ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

OPTIMIZATION OF RULE WEIGHTS AND MEMBERSHIP FUNCTIONS OF FUZZY CONTROLLER USING EXTENDED KALMAN FILTER

M.Sc. THESIS

Nasser ARGHAVANI

Department of Control Engineering

Control and Automation Engineering Programme

Thesis Advisor: Prof. Dr. Müjde GÜZELKAYA

JANUARY 2013

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

OPTIMIZATION OF RULE WEIGHTS AND MEMBERSHIP FUNCTIONS OF FUZZY CONTROLLER USING EXTENDED KALMAN FILTER

M.Sc. THESIS

Nasser ARGHAVANI
(504091132)

Department of Control Engineering

Control and Automation Engineering Programme

Thesis Advisor: Prof. Dr. Müjde GÜZELKAYA

JANUARY 2013

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

GENİŞLETİLMİŞ KALMAN FİLTRESİ İLE BULANIK KONTROLÖRÜN KURAL AĞIRLIKLARI VE ÜYELİK FONKSİYONLARININ OPTİMİZASYONU

YÜKSEK LİSANS TEZİ

Nasser ARGHAVANI
(504091132)

Kontrol Mühendisliği Anabilim Dalı

Konrol ve Otomasyon Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Müjde GÜZELKAYA

OCAK 2013

Nasser ARGHAVANI, a **M.Sc.** student of ITU **Institute of Science and Technology** student ID 504091132, successfully defended the **thesis/dissertation** entitled "**OPTIMIZATION OF RULE WEIGHTS AND MEMBERSHIP FUNCTIONS OF FUZZY CONTROLLER USING EXTENDED KALMAN FILTER**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Müjde GÜZELKAYA** İstanbul Technical University	.....
Jury Members :	**Prof. Dr. İbrahim EKSİN** İstanbul Technical University	.....
	Ass. Prof. Dr. Taner ARSAN Kadir Has University	.....

Date of Submission : 17 December 2012
Date of Defense : 18 January 2013

To my family,

FOREWORD

Firstly and foremost, I would like to express my deepest gratitude to Prof. Dr. Müjde Güzelkaya, my research supervisor, and Prof. Dr. İbrahim Eksin for their patient guidance, enthusiastic encouragement and useful critiques of this thesis.
I would also like to extend my thanks to Mortaza Aliasghary for his generously-offered assistance throughout this research work, and Mehrdad Karimzadeh Khouei for his being helpful in preparing the thesis.
Finally, I wish to thank my family, in particular my parents, for their endless love and support during my whole life.

December 2012 Nasser ARGHAVANI

TABLE OF CONTENTS

Page

ABBREVIATIONS

FLC	**:** Fuzzy Logic Controller
FLS	**:** Fuzzy Logic System
MF	**:** Membership Function
PI	**:** Proportional Integral
PD	**:** Proportional Derivative
PID	**:** Proportional Integral Derivative
FPID	**:** Fuzzy Proportional Integral Derivative
MIMO	**:** Multi Input Multi Output
SISO	**:** Single Input Single Output
EKF	**:** Extended Kalman Filter
SOPDT	**:** Second Order Plus Dead Time
CoG	**:** Center of Gravity
OS	**:** Overshoot
Ts	**:** Settling Time
Rt	**:** Rise Time
IAE	**:** Integral Absolute Error
ITAE	**:** Integral Time Absolute Error

LIST OF TABLES

Page

LIST OF FIGURES

Page

OPTIMIZATION OF RULE WEIGHTS AND MEMBERSHIP FUNCTIONS OF FUZZY PID USING EXTENDED KALMAN FILTER

SUMMARY

Mostly, it is an extremely challenging task to obtain the mathematical model of the real-life systems, and unless we have the mathematical model of the system in hand, it is somehow impossible to utilize the conventional control techniques. In addition, having a relatively accurate model of a dynamic system available does not guarantee design of a controller, since these models usually represent complicated expressions while conventional control techniques often requires some assumptions.

In such cases, intelligent approaches such as fuzzy logic provide an efficient structure to include linguistic information from human experts into numerical information, which is not possible in conventional control methods. It has been verified that fuzzy logic is the best choice for many control system applications since it mimics human control logic.

Roughly speaking, fuzzy control is ‘control with rules’. The rules are in the familiar if-then format, with the premise on the *if*-side and the conclusion on the *then*-side. The fuzzy controller has four main components: first part is the ‘rule-base’, second part is the ‘inference mechanism’, third part is the ‘fuzzification’, and last part of a fuzzy controller is the ‘defuzzification’.

Three special types of fuzzy controllers are known as PI-, PD-, and PID-type controllers which behave somehow similarly to their classical counterparts. Fuzzy PI-type control is known to be more practical than fuzzy PD-type because it is difficult for the fuzzy PD to remove steady-state error. The fuzzy PI-type control is, however, known to give poor performance in transient response for higher order processes due to the internal integration operation. Theoretically, fuzzy PID type control should enhance the performance a lot.

The overall performance of a fuzzy logic control system highly depends on the parameters of the controller. Proper choice of these parameters needs enough

theoretical and practical knowledge. In critical applications, after designing a basic fuzzy logic controller, an optimization algorithm is employed to optimize the required performance.

Parameters of fuzzy logic controller can be categorized into two main groups: *Structural parameters* and *tuning parameters.* Optimization of these parameters can be done through two approaches: derivative-based and derivative-free methods. Each method has its own advantages and disadvantages which the designer needs to choose one of the methods according to the requirements of the problem.

One of the important derivative-based optimization methods is the Kalman filtering estimation method. Kalman filter is a powerful estimator that can be utilized in an optimization problem, thus provides a tool to tune the parameters of fuzzy logic controllers. Since the relation between the inputs to the fuzzy controller and output of it is a nonlinear expression, we need to apply extended Kalman filter to optimize the parameters. In this study, extended Kalman filter is used to optimize the parameters of the rational-powered input membership functions, output singletons and rule weights of the rule base while minimize the error based evaluating function.

In order to make the problem suitable to be projected to extended Kalman filter we consider the whole system, fuzzy logic controller along with the plant, as a nonlinear relationship. Then we obtain the error model of this whole system, which parameters of the controller are considered as the states of the process and output of the system as the measurement. In the next step, we define a performance index using the absolute error or squared error between the system output and reference signals. Extended kalman filter tries to estimate the states of the process, i.e. parameters of the fuzzy logic controller while minimizing the performance index.

Firstly, we apply the extended Kalman filter to tune the membership functions parameters; namely modal points, left and right half-widths and left and right hedges of the input membership functions and positions of the output singletons. Therefore, the states to be estimated become the vector of membership function parameters and the performance index to be minimized is the squared error between the reference and output of system. The proposed online-tuned fuzzy logic controller is applied to time-delay systems with different constant time delays. As the amount of time delay increases the improvement in the unit step response becomes more tangible. Since

fuzzy PID-type controller without tuning is unable to control these time delay systems efficiently. Four performance measures, overshoot, settling time, ITAE and IAE are used to check the efficiency of the proposed method. All the measures except rise time verify the improvement in the unit step response. Rise time stays the same. Similar to the previous one, the proposed method can cope with the noise added to the control signal quite well.

Next stage was tuning the rule weights of the rule base. Therefore, the states to be estimated become the vector of rule weights and the performance index to be minimized is absolute error. Then the controller is applied to control linear systems with and without time-delay. The results show improvement in the unit step response of the system in comparison with controller without tuning algorithm. Overshoot and settling time decreased considerably while rise time stays the same. Moreover, decrease in the value of ITAE and IAE shows the efficiency of the proposed online tuning method. Also, extended Kalman filter-based optimization could successfully handle the artificially added noise to the control signal.

GENİŞLETİLMİŞ KALMAN FİLTRESİ İLE BULANIK KONTROLÖRÜN KURAL AĞIRLIKLARI VE ÜYELİK FONKSİYONLARININ OPTİMİZASYONU

ÖZET

Genellikle, gerçek bir sistemin matematiksel modelinin çıkarılması zordur ve matematiksel modellerin elde edilememesi halinde, klasik kontrol yöntemlerinin uygulanması imkansız hale gelir. Ayrıca, elde edilen modellerde çok karmaşık matematik denklemlerin bulunması ve klasik kontrolörlerin bazı varsayımlara ihtiyaç duyması nedeniyle, doğru bir matematiksel model elde edilmesi, kontrolörün tasarımını garantiye almamaktadır.

Bu gibi durumlarda, bulanık mantık gibi akıllı yöntemler, klasik yöntemlerde mümkün olmayan bir şekilde, uzmandan gelen dilsel bilgileri sayısal bilgilere çevirerek etkili yapıların ortaya çıkmasını sağlarlar. Bulanık mantık, insana ait kontrol mantığını taklit edebilmesi nedeniyle, kontrol sistem uygulamalarında kullanılabilecek en iyi yöntemlerden biridir. Bulanık mantık tabanlı kontrol için, yüzeysel olarak, "kurallarla kontrol etmek" sözü kullanılabilir. Bu koşullar "eğer-o halde" biçiminde olup kuralın öncül kısmı "eğer" tarafında ve sonuç kısmı "o halde" tarafındadır. Bulanık kontrolör dört kısımdan oluşmaktadır: birinci kısım kural tabanı, ikinci kısım çıkarım mekanizması, üçüncü kısım bulanıklaştırma ve son kısım durulaştırmadır.

Bulanık kontrolörün üç özel tipi, PI-, PD- ve PID- tipi bulanık kontrolörler olup bunlar klasik PI, PD ve PID kontrolörlerine çok yakın davranışlar sergiledikleri gösterilmiştir. PD-tipi kotrolör ile sürekli hal hatasını ortadan kaldırmak zor olduğundan, pratikte, bulanık PI-tipi kontrolörler, bulanık PD-tipi kontrolörlere oranla daha fazla kullanılır. Ancak, bulanık PI-tipi kontrolörlerin yüksek mertebeli sistemlerin geçici hal yanıtları üzerindeki başarımı zayıftır. Teorik olarak, bulanık PID-tipi kontrolör yukarıda anlatılan sorunları gidermektedir.

Bir bulanık kontrol sisteminin genel başarımı, kontrolör parametrelerine çok bağlıdır. Bu parametrelerin doğru seçilmesi için yeterli teorik ve pratik bilgiler gerekmektedir. Kritik uygulamalarda, bulanık kontrolörün temel tasarımı yapıldıktan sonra, başarımı en iyi hale getirmek üzere çeşitli parametrelerin belirlenmesi amacıyla optimizasyon yöntemleri kullanılır.

Bulanık mantık kontrolörünün tasarım parametreleri iki temel gruba ayrılmaktadır: *Yapısal parametreler* ve *ayar parametreleri*. Yapısal parametreler, giriş/çıkış değişkenleri, bulanık dilsel kümeler, üyelik fonksyonları, bulanık kurallar, çıkarım ve durulaştırma mekanizmalarından oluşmaktadır. Ayar parametreler ise, giriş/çıkış ölçekleme çarpanları ve üyelik fonksiyonlarının parametreleridir.

Yukarıda adı geçen parametrelerin optimizasyonu iki farklı yaklaşımla yapılabilir. Bunlar, "türev temelli" ve "türev temelli olmayan" yaklaşımlardır. Türev temelli olmayan yöntemlerde kontrolör parametreleri açısından başarım ölçütünün türevi gerekmez. Bu nedenle, türev temelli yöntemlere göre dayanaklılardır. Bu yöntemlerle global minimum bulunduğundan değişik üyelik ve amaç fonksyonlarına uygulanabilirler. Türev temelli olmayan yöntemler, türev temelli yöntemlere göre göre daha yavaştır. Ancak türev alındığı için, özel yapıda üyelik ve amaç fonksyonları ve çıkarım mekanizmalarına uygulanabilirler.

Önemli türev temelli optimizasyon yöntemlerinden biri "Kalman filtreleme" yöntemidir. Kalman filtresi, optimizasyon problemlerinde güçlü bir araç olduğundan, bulanık mantık kontrolörlerin parametrelerinin ayarlamasında da kullanılabilir. Kalman filtresinin doğrusal olmayan modeller için doğrusallaştırmaya yardımıyla uygulanan biçimine "Genişletilmiş Kalman Filtresi" adı verilir. Bulanık kontrolörlerde giriş ve çıkış ifadeleri arasındaki ilişki doğrusal olmadığından genişletilmiş Kalman filtresinin uygulanmasına ihtiyaç vardır.

Bu tezde, ilk olarak, genişletilmiş Kalman filtresi kullanılarak, bulanık PID-tipi kontrolöre ilişkin rasyonel kuvvetli giriş üyelik fonksiyonlarının ve tekli çıkış üyelik fonksiyonlarınn çevrim içi parametre optimizasyonu yapılmıştır. Ayrıca, bulanık PID-tipi kontrolörlerin yapısal parametrelerinden olan kural ağırlık katsayıları, yine genişletilmiş Kalman filtresi kullanılarak çevrim içi belirlenmiştir. Başarım ölçütü, kontrol sisteminin referans ve çıkış işaretleri arasındaki farkın mutlak veya karesel değerleri olarak tespit edilmiştir. Problemin genişletilmiş Kalman filtresine uygun

hale getirilmesi için kontrolör ve kontrol edilen sistem bir arada ele alınmış, hata modeli çıkarılmıştır. Kalman filtresi açısından, optimal olarak belirlenmek istenen parametreler, filtrenin kestireceği durumlar ve kontol sisteminin çıkışı, ölçüm olarak ifade edilmiştir. Bulanık mantık kontrolörünün parametreleri hata üzerinden tanımlanmış başarım ölçütü en küçüklenmek üzere genişletilmiş Kalman filtresi ile kestirilmiştir.

İlk olarak, bulanık PID-tipi kontrolöre ilişkin rasyonel kuvvetli giriş üyelik fonksiyonlarının ve tekli çıkış üyelik fonksiyonlarının parametre optimizasyonu yapılmıştır. Giriş üyelik fonksiyonların merkez noktası, sağ ve sol bacakları, sağ ve sol biçimlendiricileri ve tekli çıkış üyelik fonksiyonların yeri ayarlanmıştır. Davranış ölçütü kontrol sistem hatasının karesi olarak tanımlanmıştır. Önerilen yöntem, benzetim çalışmaları ile zaman gecikmesi olan sistemler üzerinde denenmiştir. Yöntemin sistem başarımı üzerindeki etkisi, aşım, oturma zamanı, ITAE ve IAE ölçütlerine göre parametre ayarlaması bulunmayan PID-tipi bulanık kontrolör ile karşılaştırılmıştır. Önerilen yöntemin sistem başarımı üzerindeki olumlu etkisinin gecikme arttıkça daha fazla olduğu görülmüştür. Ayrıca kontrol işaretine gürültü eklenerek yöntemin gürültü üzerindeki başarılı etkisi de test edilmiştir.

Kural tabanı, bulanık kontrolörlerin en etkili yapısal parametrelerindendir. Kural ağırlıklarının doğru bir biçimde belirlenebilmesi ile kural tabanının sistem başarımı üzerindeki artırılabilir. Bu nedenle, tezin sonraki aşamasında, genişletilmiş Kalman filtresi ile kural ağırlıklarının belirlemesinde kullanılmıştır. Filtrenin kestirmesi gereken durumlar, kural ağırlıkları vektörü ve kontrol sistem çıkışı filtrenin ölçüm değeri olarak kullanılmıştır. Başarım ölçütü mutlak hata olarak belirlenmiştir. Elde edilen kural ağırlık ayarlamalı bulanık kontrolör, zaman gecikmesi olan ve olmayan doğrusal sistemler üzerinde benzetim çalışmaları karşılaştırmalı olarak denenmiştir. Aşım, yerleşme zamanı, ITAE ve IAE değerleri önemli ölçüde azalmış ama yükselme zamanı sabit kalmıştır. Ayrıca yöntemin gürültü üzerindeki etkisi de test edilmiştir.

1. INTRODUCTION

Although proposing a lot of research and a high number of different solutions, most industrial control systems are still based on proportional integral derivative (PID) controllers control. It is believed that 90% to 99% of industrial processes make use of the famous PID controllers. Some reasons of this amount of use can be:

a. Design of PID controller is simple.

b. There is a certain relationship between the parameters of the PID controller with the response of the system.

c. In the last decades, a great deal of tuning approaches for optimizing the PID controller parameters has been proposed.

However, PID controller cannot be a general approach for all the control problems. Time –delay, nonlinear, or time-variant systems can be the example of systems that classical PID controller faces difficulty coping with. Fuzzy control, being included in the area of artificial intelligence and control engineering, is thought to be a solution for the problem.

Following the first fuzzy control application carried out by Mamdani [1], fuzzy logic is utilized quite often in control problems. During the past few decades, there are many successful applications with fuzzy logic controllers (FLC) in industry. They have been reported to be successfully used for a number of complex and non-linear processes. Moreover, the experience has shown that, fuzzy control may often be a preferred method of designing controllers for dynamic systems even if traditional methods can be used [2].

Between the various types of fuzzy logic controllers, just like the widely used conventional PID controllers in process control systems, PID-type FLCs are most common and practical.

The research for improving FLC performance spreads over number of areas. But simply, we can categorize the design parameters within two groups:

a. Structural parameters

b. Tuning parameters

There have been numerous methods to optimize these parameters to improve the performance of FLCs. A lot of heuristic and non-heuristic tuning algorithms for the adjustment of scaling factors of fuzzy controllers have been presented in literature [3, 4, 5, 6]. The literature also includes some methods to tune the membership functions and fuzzy rules of the fuzzy controllers. Juang, Chang, and Huang [7], tuned the membership functions by means of genetic algorithm (GA) using a fitness function to improve the system performance. Ahn and Truong [8] used a robust extended Kalman filter to tune the input membership functions and the weight of the controller output during the system operation process. Teng, Xiang, Wang, and Wu [9] proposed a genetic weighted fuzzy rule based system in which the parameters of membership functions including position and shape of the fuzzy rule set and weights of the rules are evolved using a genetic algorithm. Genc, Yesil, Eksin, Guzelkaya, and Tekin [10] proposed a fuzzy rule base shifting scheme for systems with time to improve system performance.

Another rough classification of parameter optimization is based on derivative with respect to the parameters; namely derivative-free and derivative-based methods. Derivative-free methods have the advantage that they do not require the derivative of the objective function with respect to the membership function parameters. They are more robust than derivative-based methods with respect to finding a global minimum and with respect to their applicability to a wide range of objective functions and membership function forms. However, they typically tend to converge more slowly than derivative-based methods. Derivative-based methods have the advantage of fast convergence, but they tend to converge to local minima. In addition, due to their dependence on analytical derivatives, they are limited to specific objective functions, specific types of inference, and specific types of membership functions [11].

Some of the derivative-free methods are genetic algorithm [12], neural network [13], evolutionary programming [14], geometric methods [15], fuzzy equivalence relations [16], heuristic methods [17], etc. The most common approaches of derivative-based method are gradient descent [18, 19], least squares [20], back propagation [21], and Kalman filtering [19, 22].

Most of the practical processes under automatic control are nonlinear higher order systems and may have considerable dead time. Higher order systems can be modeled by a first order counterpart as soon as convenient approximations are carried, but control action is unavoidably delayed in a process with dead time. For this reason, dead time is recognized as the most difficult dynamic element naturally occurring in physical systems. Therefore, any useful technique of designing a control system must be capable of dealing with dead time. Conventional PID (or PI) controllers are frequently short in managing systems with dead time. To have a satisfactory performance the controller output or process input should be a nonlinear function of error and change of error. FLCs try to incorporate this nonlinearity by a limited number of if-then rules. As a result FLCs are used more commonly as the time delay or nonlinearity is the matter of concern.

Noise and disturbance are known to be unwanted signals which are ubiquitous in all environments, and all real systems are exposed to such signals all the time. The matter of noise and disturbance has been the hot topic of research where a lot of noise reduction schemes have been proposed in the literature. One solution for this difficulty has been making use of estimation methods, such as H_∞ estimation method or Kalman filtering approach. Fuzzy logic, with its intrinsic intelligence feature, is capable of dealing with noise and disturbance to some extent. However, combining fuzzy logic controller with an estimation method, e.g. extended Kalman filter, can provide an efficient method to cope with this trouble.

Extended Kalman filter (EKF) has been used with fuzzy logic in various ways. For instance, Kalman filters have been used to extract fuzzy rules from a given rule base [23]. Kalman filters have also been used to optimize the output function parameters of Takagi–Sugeno–Kang fuzzy systems [24]. Ahn and Truong [8] used a robust extended Kalman filter to tune the input membership functions and the weight of the controller output during the system operation process.

This thesis is focused on the subject of tuning fuzzy logic controller, particularly fuzzy PID-type controller, by using extended Kalman filter. Previously EKF had been applied to optimize the parameters of triangular input membership functions of fuzzy tuners [8], however in this study, we extended membership functions to rational-powered membership functions and also fuzzy block is used as the main controller. On the other hand, the main contribution of this work is online tuning of

the fuzzy rule weights of fuzzy PID-type controller via EKF. Then the EKF-based fuzzy PID controller is applied to linear systems with and without time delay. Four performance measures are defined to check the efficiency of the proposed controller. In addition, artificial noise is added to the signal control to simulate the real environment. The proposed controller can successfully cope with the noise in comparison with the two other controllers (i.e. non-tuned fuzzy PID and Gradient descent based fuzzy PID).

The rest of the thesis study is divided into 6 sections. In section 2, firstly the concept of fuzzy logic is discussed. Then we extend it to fuzzy control, in which structure of fuzzy PID-type and some tuning methods are explained. The mathematical background, theory and algorithm of Kalman filter and extended Kalman filter will be given in section 3. In section 4, the application of the extended Kalman filter to membership function parameter tuning is explained and the related simulations will be given. Optimization of rule weights via the proposed approach and corresponding simulations are given in section 5. Finally, further discussion and conclusion is presented in section 6.

2. FUZZY LOGIC AND FUZZY CONTROL

Conventional control techniques are generally requires mathematical models of systems to design a controller. On the other hand, most of real-life systems' mathematical models are not very easy to obtain. Therefore all the information; numerical and linguistic information should be investigated throughout the modeling stage.

Even if a relatively accurate model of a dynamic system can be developed, it is generally too complex to use in controller development, especially for many conventional control design procedures that require some assumptions.

In such cases, fuzzy control provides an efficient structure to include linguistic information from human experts into numerical information. This is not possible in conventional control techniques. In this type of case, fuzzy controllers can be preferred.

The concept of Fuzzy Logic was proposed by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership. This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time. Professor Zadeh taught that people do not require precise, numerical information input, and yet they are capable of highly adaptive control. If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement.

As the complexity of a system increases, it becomes more difficult and sometimes impossible to make a precise model.

Fuzzy logic is also considered as a problem-solving control system methodology. It can be implemented in hardware, software or a combination of both.

Fuzzy logic was conceived as a better method for sorting and handling data but has verified to be a best choice for many control system applications since it mimics

human control logic. It can be built into anything from small, hand-held products to large automated process control systems. It uses an imprecise but very helpful language to deal with input data more like a human operator. The approach to control problems is generally to mimic how an operator would make decisions, only much faster.

Fuzzy controllers are used to control consumer products, such as washing machines, video cameras, and as well as industrial processes [3].

2.1 Fuzzy logic

2.1.1 Fuzzy sets

Fuzzy sets are a further development of mathematical set theory, first studied formally by the German mathematician Georg Cantor (1845-1918). It is possible to express most of mathematics in the language of set theory, and researchers are today looking at the consequences of 'fuzzifying' set theory, resulting in, for example, fuzzy logic, fuzzy numbers, fuzzy intervals, fuzzy arithmetic, and fuzzy integrals. Fuzzy logic is based on fuzzy sets, and with fuzzy logic a computer can process words from natural language, such as 'small', 'large', and 'approximately equal'.

Given a collection of objects U, a fuzzy set A in U is defined as a set of ordered pairs

$$A \stackrel{\text{def}}{=} \{ \langle x, \mu_A(x) \rangle \mid x \in U \} \qquad \textbf{(2.1)}$$

where $\mu_A(x)$ is called the membership function for the set of all objects x in U. The membership function relates to each x a membership grade $\mu_A(x)$, a real number in the closed interval $[0, 1]$. Members of a fuzzy set are taken from a universe of discourse, or universe for short [4].

2.1.2 Membership function

There are two alternative ways to represent a membership function: continuous or discrete. A continuous fuzzy set A is defined using a continuous membership function $\mu_A(x)$. A trapezoidal membership function is a piecewise linear, continuous function, controlled by four parameters $\{a, b, c, d\}$ [25]

$$\mu_{Trapezoid}(x; a, b, c, d) = \begin{cases} 0 & , x \leq a \\ \dfrac{x-a}{b-a} & , a \leq x \leq b \\ 1 & , b \leq x \leq c \\ \dfrac{d-x}{d-c} & , c \leq x \leq d \\ 0 & , d \leq x \end{cases} \quad \textbf{(2.2)}$$

The parameters $a \leq b \leq c \leq d$ define four breakpoints, designed as follows: left foot point (a), left shoulder point (b), right shoulder point (c), and right foot point (d). Figure 2.1 (a) illustrates a trapezoidal membership function.

A *triangular* membership function is piecewise linear, and derived from the trapezoidal membership function by merging the two shoulder points into one, that is, setting $b = c$, as in Figure 2.1 (b).

Smooth, differentiable versions of the trapezoidal and triangular membership functions can be obtained by replacing the linear segments corresponding to the intervals $a \leq x \leq b$ and $c \leq x \leq d$ by a nonlinear function, for instance, a half period of a cosine function,

$$\mu_{STrapezoid}(x; a, b, c, d) = \begin{cases} 0 & , x \leq a \\ \dfrac{1}{2} + \dfrac{1}{2}\cos\left(\dfrac{x-b}{b-a}\pi\right) & , a \leq x \leq b \\ 1 & , b \leq x \leq c \\ \dfrac{1}{2} + \dfrac{1}{2}\cos\left(\dfrac{x-c}{d-c}\pi\right) & , c \leq x \leq d \\ 0 & , d \leq x \end{cases} \quad \textbf{(2.3)}$$

We call it *STrapezoid*, for '*smooth trapezoid*' *or* '*soft trapezoid*'. Figures 2.1 (c-d) illustrate the smooth membership functions. Other possibilities exist for generating smooth trapezoidal functions, for example, Gaussian, generalized bell, and sigmoidal membership functions [25].

Discrete fuzzy sets are defined by means of a discrete variable x_i $(i = 1, 2, \dots)$. A discrete fuzzy set A is defined by ordered pairs,

$$A = \{ \langle x_1, \mu(x_1) \rangle, \langle x_2, \mu(x_2) \rangle, \dots | x_i \in U, i = 1, 2, \dots \} \quad \textbf{(2.4)}$$

Each membership value $\mu(x_i)$ is an evaluation of the membership function μ at a discrete point x_i in the universe U, and the whole set is a collection, usually finite, of pairs $\langle x_i, \mu(x_i)\rangle$.

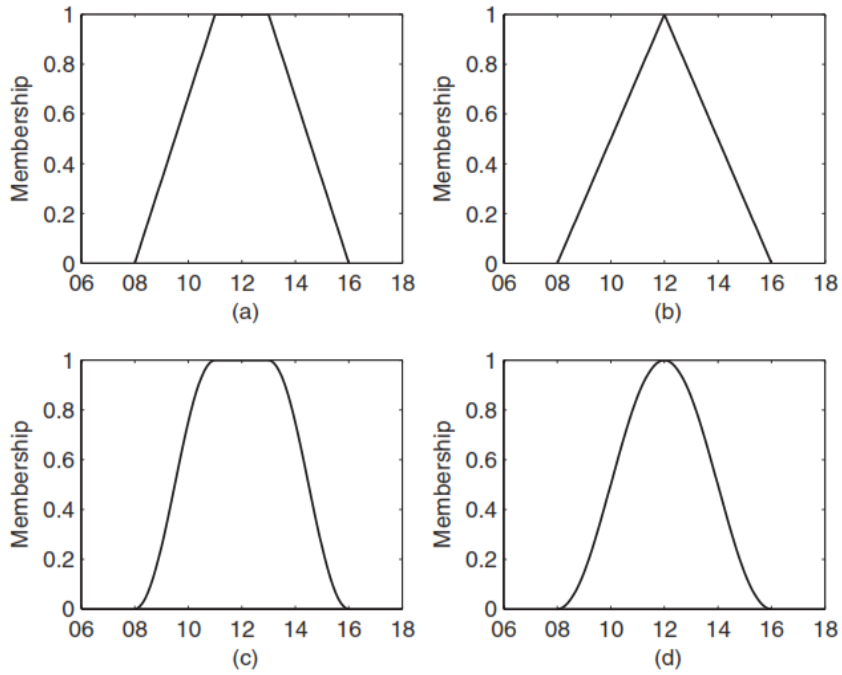


Figure 2.1 : (a) trapezoidal, (b) triangular, (c) smooth trap., and (d) smooth trian.

2.1.3 Fuzzy set operations

Let A and B be fuzzy sets defined on a mutual universe U. The fuzzy union of A and B is

$$A \cup B \equiv \{\langle x, \mu_{A\cup B}(x)\rangle \,|x \in U \; and \; \mu_{A\cup B}(x) = \max(\mu_A(x), \mu_B(x))\} \quad \textbf{(2.5)}$$

The fuzzy intersection of A and B is

$$A \cap B \equiv \{\langle x, \mu_{A\cap B}(x)\rangle \,|x \in U \; and \; \mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))\} \quad \textbf{(2.6)}$$

The fuzzy complement of A is

$$\bar{A} \equiv \{\langle x, \mu_{\bar{A}}(x)\rangle \,|x \in U \; and \; \mu_{\bar{A}}(x) = 1 - \mu_A(x)\} \quad \textbf{(2.7)}$$

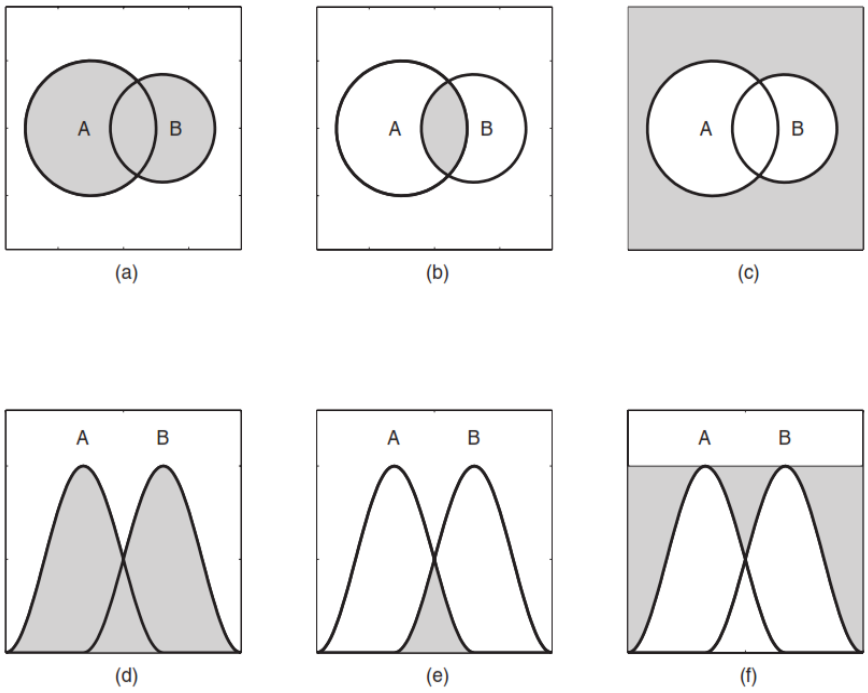


Figure 2.2 : Set operation, classical Venn diagrams and fuzzy equivalents.

2.2 Fuzzy control

Roughly speaking, fuzzy control is 'control with rules'. A fuzzy controller can include empirical rules, and that is especially useful in operator-controlled plants. Consider the typical fuzzy controller:

If *error* is Neg and *change in error* is Neg then *control* is NB.

If *error* is Neg and *change in error* is Zero then *control* is NM.

The rules are in the familiar if-then format, with the premise on the *if*-side and the conclusion on the *then*-side. The premise value 'Neg' is a *linguistic term* short for the word 'negative', the conclusion value 'NB' stands for 'negative big' and 'NM' for 'negative medium'. The collection of rules is a *rule base*. A computer can execute the rules and compute a control action depending on the measured inputs *error* and *change in error*.

In the *direct control* scheme in Figure 2.3 the fuzzy controller is in the forward path of a feedback control system. The plant output y is compared with a reference *Ref*, and if there is a deviation $e = Ref - y$, the controller takes action according to the control strategy embedded in the rule base.

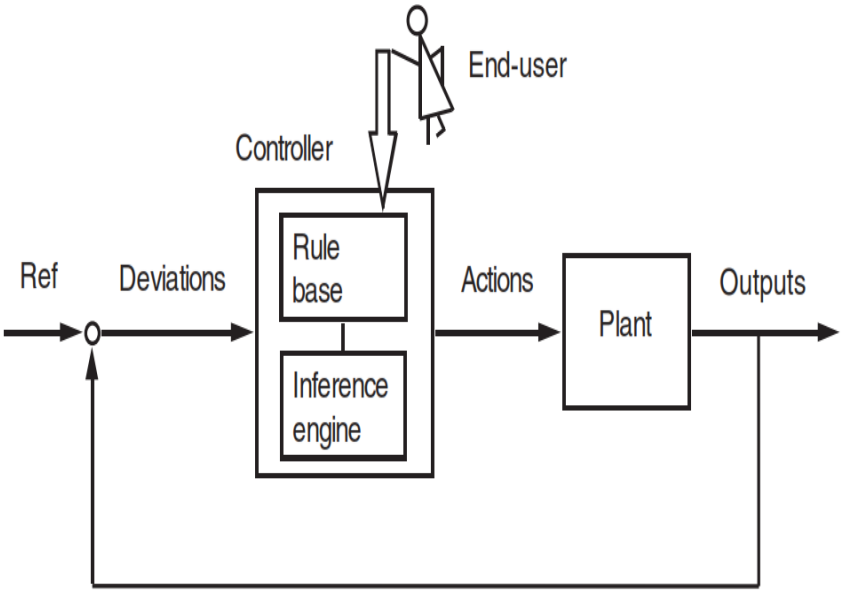


Figure 2.3 : Direct control.

There are at least four main sources for finding control rules [26]:

- *Expert experience and control engineering knowledge.* The most common approach is to question experts or operators with a carefully organized questionnaire.
- *Based on the operator's control actions.* Observing an operator's control actions or examining a logbook may reveal fuzzy *if –then* rules of input–output relationships.
- *Based on a fuzzy model of the plant.* A linguistic rule base may be viewed as an inverse model of the controlled plant. Thus the fuzzy control rules can perhaps be obtained by inverting a fuzzy model of the plant, if fuzzy models of the open and closed-loop systems are available [26]. This method is restricted to low order systems. Another approach is *fuzzy identification* [26].
- *Based on learning.* The self-organizing controller is an example of a controller that finds the rules itself. It is an example of the more general control scheme: model reference adaptive control.

2.2.1 Internal Structure of Fuzzy Controller

In the Figure 2.4, the controller is between a pre-processing block and a post-processing block.

2.2.1.1 Pre-processing block

Let us assume the inputs are *crisp* measurements from measuring equipment, rather than linguistic. A pre-processor, the first block in Figure 2.4, conditions the measurements before they enter the controller. Examples of pre-processing are

- quantization in connection with sampling or rounding to integers;
- normalization or scaling onto a particular, standard range;
- filtering in order to remove noise;
- averaging to obtain long-term or short-term tendencies;
- a combination of several measurements to obtain key indicators; and
- Differentiation and integration, or their approximations in discrete time.

2.2.1.2 Post-processing block

If the inferred control value is defined on a standard universe, it must be scaled to engineering units, for instance: volts, meters, or tons per hour. An example is the scaling from the standard universe $[-1, 1]$ to the physical units $[-10, 10]$ volts. The post-processing block contains an output gain that can be tuned.

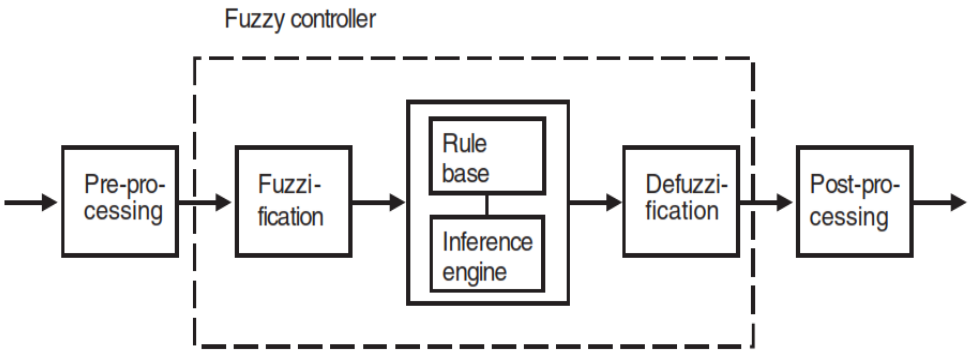


Figure 2.4 : Building blocks of a fuzzy controller.

The fuzzy controller between the above mentioned blocks has four main components: first part is the ‘rule-base’ where the knowledge is held in the form of a set of rules. Second part is the ‘inference mechanism’ where evaluations are made, which control rules are related at that time and then decides what the input to the plant should be given. Third part is the ‘fuzzification’ simply modifies the inputs so that they can be interpreted and compared to the rules in the rule-base. Last part of a

fuzzy controller is the ‘defuzzification’ that converts the fuzzy outputs decided by the inference mechanism into the crisp inputs to the plant.

2.2.1.3 Fuzzification

Fuzzification is the first step in the fuzzy inference process. This involves a transformation of crisp inputs into fuzzy inputs. Crisp inputs are exact inputs measured by sensors and passed into the control system for processing, such as temperature, pressure, rpm’s, etc. Each crisp input that is to be processed by the Fuzzy Inference Unit has its own group of membership functions or sets to which they are transformed. This group of membership functions exists within a universe of discourse that holds all relevant values that the crisp input can possess.

2.2.1.4 Rule Base

The rules may use several variables both in the condition and the conclusion of the rules. The controllers can therefore be applied to both multi-input-multi-output (MIMO) problems and single-input-single-output (SISO) problems. The typical SISO problem is to regulate a control signal based on an error signal. The controller may actually need both the error, the change in error, and the integrated error as inputs, because in principle all three are formed from the error measurement. To simplify, the control objective is to regulate some process output around a prescribed set-point or reference.

Basically a linguistic controller contains rules in the *if-then* format.

1. If *error* is Negative and *change in error* is Negative then the output is Negative Big.
2. If *error* is Negative and *change in error* is Zero then the output is Negative Medium.
3. If *error* is Negative and *change in error* is Positive then the output is Zero.
4. If *error* is Zero and *change in error* is Negative then the output is Negative Medium.

2.2.1.5 Inference Mechanism

The inference mechanism has two basic tasks: one is to determining the degree, to which each rule is relevant to the current situation. Inputs that passed through the

fuzzification stage are evaluated for each rule in the rule base. Depending on the inputs, one or more that one rule may be satisfied.

Other task is deciding the control action using the current inputs and the information in the rule-base. The output of the inference mechanism becomes the input of the defuzzification stage.

The following figures illustrate the different sorts of inference mechanism; Mamdani and Sugeno Types, which are the most popular reasoning schemes.

- Mamdani Inference mechanism: two types of this reasoning scheme are shown below:

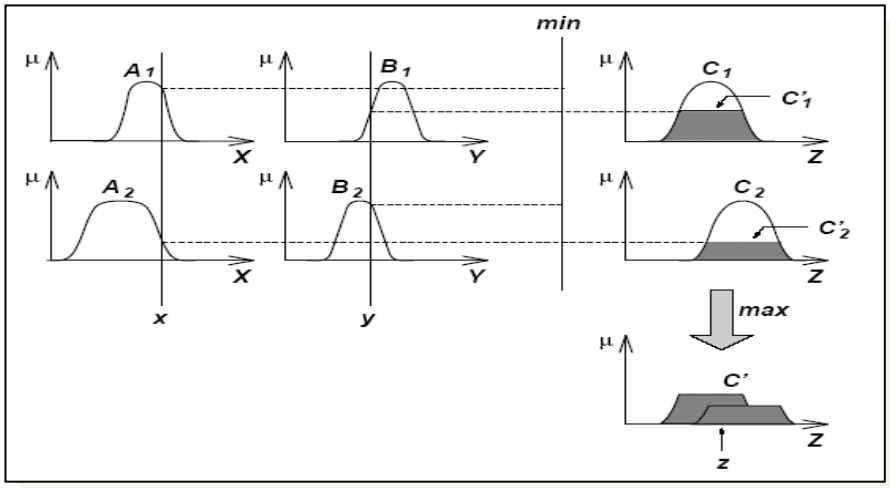


Figure 2.5 : Max-min composition.

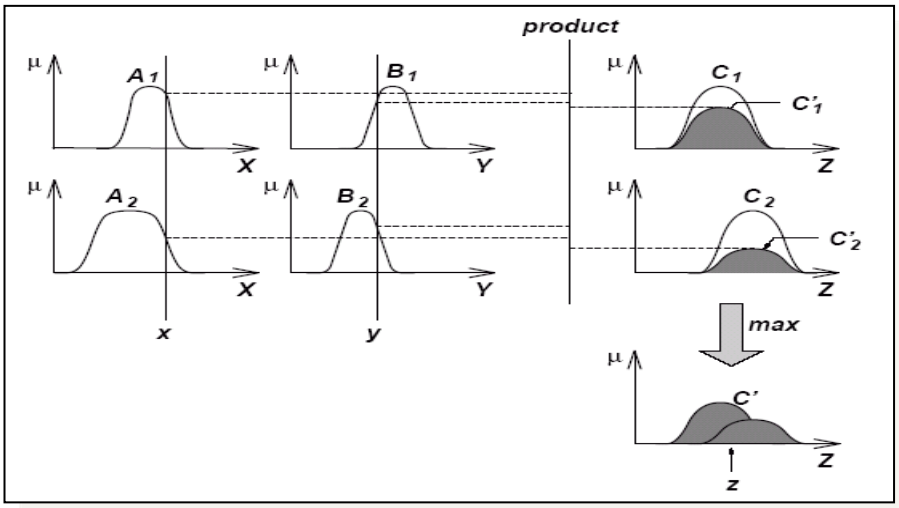


Figure 2.6 : Max-product composition.

- Sugeno Inference Mechanism: Figure 2.7 illustrates reasoning scheme when the consequent part of *if-then* rules is a function.

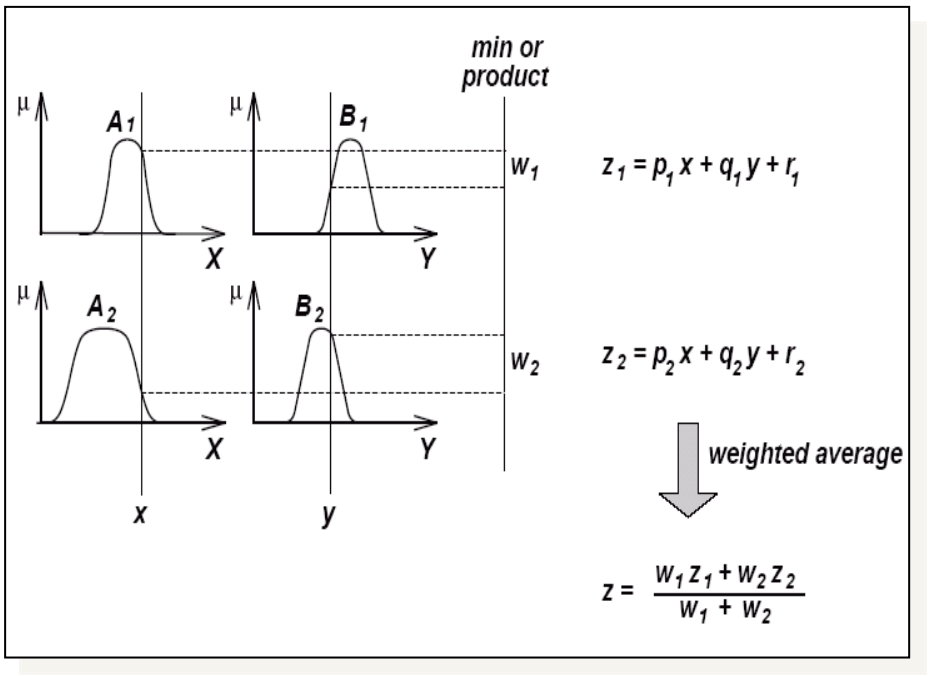


Figure 2.7 : Sugeno reasoning scheme.

2.2.1.6 Defuzzification

The output of the inference mechanism is the input of the defuzzification stage. The decided control action which has fuzzy values is converted into the crisp values with the help of defuzzification methods. There are many methods to defuzzify the fuzzy values. The 'centroid' method is very popular, in which the 'center of mass' of the result provides the crisp value. Another approach is the 'height' method, which takes the value of the biggest contributor.

- Center of Gravity (CoG): the most commonly used method to defuzzify the fuzzy values is CoG, which calculates the center of gravity of the finally reached areas.

$$y^* = \frac{\int_U y\mu_u(y)dy}{\int_U \mu_u(y)dy} \qquad \textbf{(2.8)}$$

- Maximum method: in this method firstly the highest area is obtained, then according to that area three different methods can be used; smallest of that area, largest of the area or the mean of the area.

$$y_1^* = \inf\{y \in Y \mid \mu_Y(y) = hgt(Y)\} \quad \textbf{(2.9)}$$

$$y_3^* = \sup\{y \in Y \mid \mu_Y(y) = hgt(Y)\} \quad \textbf{(2.10)}$$

$$y_2^* = \frac{y_1^* + y_3^*}{2} \quad \textbf{(2.11)}$$

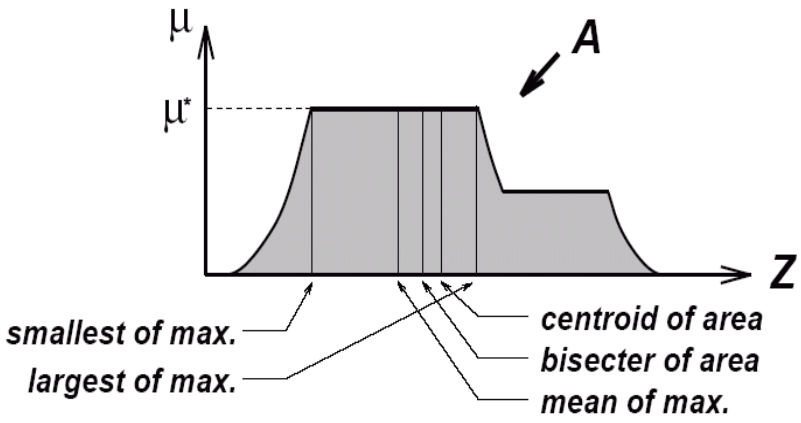


Figure 2.8 : Diffuzification methods.

2.2.2 PID-Type Fuzzy Controller Structure

Most of the applications of fuzzy control adopt the error and the change rate of error of the output in the premise of the rule-base. It has also been proved that such a fuzzy controller is approximately equal to a linear PD controller for sufficiently large number of rules [22].

$$u = \alpha U = \alpha(A + PK_1 e + DK_2 \dot{e}) = \alpha A + \alpha PK_1 e + \alpha DK_2 \dot{e} \quad \textbf{(2.12)}$$

Since the mathematical models of most industrial process systems are of type 0, obviously there would exist a steady-state error if they are controlled by this kind of fuzzy controller. If we want to eliminate the steady-state error of the control system, we can imagine substituting the input $\dot{e}$ (the change rate of error or the derivative of error) of the fuzzy controller with the integration of error. This will result the fuzzy controller behaving like a parameter time-varying PI controller, thus the steady-state error is expelled by the integration action. However, a PI type fuzzy controller will have a slow rise time if the P parameters are chosen small, and have a large

overshoot if the P or I parameters are chosen large. So there may be the time when one wants to introduce not only the integration control but the derivative control to the fuzzy control system, because the derivative control can reduce the overshoot of the system's response so as to improve the control performance. Of course this can be realized by designing a fuzzy controller with three inputs, error, and the change rate of error and the integration of error. However, these methods will be hard to implement in practice because of the difficulty in constructing fuzzy control rules. Usually fuzzy control rules are constructed by summarizing the manual control experience of an operator who has been controlling the industrial process skillfully and successfully. The operator intuitively regulates the executor to control the process by watching the error and the change rate of the error between the system's output and the set-point value. It is not the practice for the operator to observe the integration of error. Moreover, adding one input variable will greatly increase the number of control rules, the constructing of fuzzy control rules are even more difficult task and it needs more computation efforts. One way is to have an integrator serially connected to the output of the fuzzy controller as shown in Figure 2.9.

$$u = \beta \int Udt = \beta \int (A + PK_1 e + DK_2 \dot{e})dt$$
$$= \beta At + \beta K_2 De + \beta K_1 P \int edt \qquad \textbf{(2.13)}$$

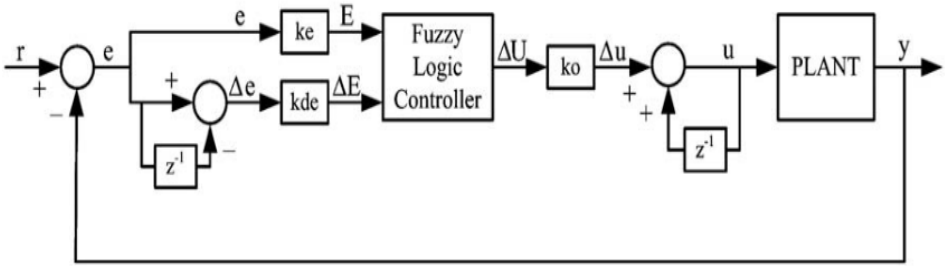


Figure 2.9 : The basic control block diagram illustration with PI type FLC.

Hence the fuzzy controller becomes a parameter time-varying PI controller, its equivalent proportional control and integral control components are $\beta K_2 D$ and $\beta K_1 P$ respectively. This fuzzy controller is called as the PI type fuzzy controller [27].

Fuzzy PI-type control is known to be more practical than fuzzy PD-type because it is difficult for the fuzzy PD to remove steady-state error. The fuzzy PI-type control is,

however, known to give poor performance in transient response for higher order processes due to the internal integration operation. Theoretically, fuzzy PID type control should enhance the performance a lot. It should be pointed out that, for fuzzy PID controllers, normally a 3-D rule base is required. This is difficult to obtain since 3-D information is usually beyond the sensing capability of a human expert. To obtain proportional, integral and derivative control action all together, it is intuitive and convenient to combine PI and PD actions together to form a fuzzy PID controller.

The formulation of PID-type FLC can be achieved either by combining PI- and PD-type FLCs with two distinct rule-bases or one PD-type FLC with an integrator and a summation unit at the output. In the latter kind of PID-type FLC, the number of scaling factors is decreased compared to the PID-type FLC formed by combining PI- and PD type FLCs with two distinct rule-bases [28].

Figure 2.10 shows the structure of a standard fuzzy PID-type controller. The output of the fuzzy PID-type controller is given by

$$u = \alpha U + \beta \int U dt \tag{2.14}$$

where U is the output of the FLC. It has been shown in [27] and [29] that the fuzzy controllers with product–sum inference method, center of gravity defuzzification method and triangular uniformly distributed membership functions for the inputs and a crisp output, the relation between the input and the output variables of the FLC is given by

$$U = A + PE + D\dot{E} \tag{2.15}$$

where $E = K_e e$ and $\dot{E} = K_d \dot{e}$. The same result is shown to be valid for the minimum inference engine in [30]. Therefore from Equations 2.14 and 2.15 the controller output is obtained as

$$u = \alpha A + \beta A t + \alpha K_e P e + \beta K_d D e + \beta K_p P \int e dt + \alpha K_d D \dot{e} \tag{2.16}$$

Thus, the equivalent control components of the PID type FLC are obtained as follows:

Proportional gain: $\alpha K_e P + \beta K_d D$

Integral gain: $\beta K_p P$

Derivative gain: $\alpha K_d D$

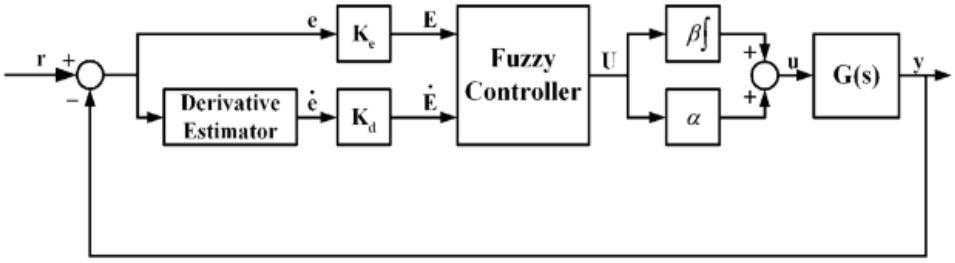


Figure 2.10 : Closed-loop control structure for PID-type FLC.

2.2.3 Gradient Descent-based Optimization

Gradient Descent Optimization method is a relatively simple derivative-based optimization approach which provides satisfactory results. In analysis of most derivative-based optimization methods, it provides a quite accetable tool for the purpose of comparison. In this study, we introduce this method briefly for the same purpose.

The gradient descent method can be applied to optimize the parameters of membership functions and also rule weights as follows:

$$c_{ij}(k+1) = c_{ij}(k) - \eta \frac{\partial E(k)}{\partial c_{ij}} \quad \textbf{(2.17)}$$

$$a_{ij}^{+/-}(k+1) = a_{ij}^{+/-}(k) - \eta \frac{\partial E(k)}{\partial a_{ij}^{+/-}} \quad \textbf{(2.18)}$$

$$b_{ij}^{+/-}(k+1) = b_{ij}^{+/-}(k) - \eta \frac{\partial E(k)}{\partial b_{ij}^{+/-}} \quad \textbf{(2.19)}$$

$$\omega_{ij}(k+1) = \omega_{ij}(k) - \eta \frac{\partial E(k)}{\partial \omega_{ij}} \quad \textbf{(2.20)}$$

where η is the gradient descent step size.

3. **KALMAN FILTER**

It has been more than four decades since Kalman introduced his systematic approach to linear filtering based on the method of least-squares [31]. Although his original journal article was difficult to read and understand, the results of the paper were applied immediately in many different fields by individuals with a variety of backgrounds because the filtering algorithm actually worked and was easy to implement on a digital computer. People were able to apply Kalman filtering without necessarily understanding or caring about the intricacies of its derivation. Because of the case of implementation of the original recursive digital Kalman filter, engineers and scientists were able to find out immediately that this new filtering technique was often much better than existing filtering techniques in terms of performance. Both performance improvements and ease of implementation rather than analytical elegance made the Kalman filter popular in the world of application. However, the Kalman filter was usually much more computationally expensive than existing filtering techniques, which was an issue in many cases for the primitive computers that were available at that time. In addition to improved performance, this new filtering technique also provided a systematic approach to many problems, which was also an improvement over some of the ad hoc schemes of the day. Today, because of the popularity and proliferation of Kalman filtering, many individuals either do not know (or care) about any other filtering techniques. Some actually believe that no filtering took place before 1960.

With the possible exception of the fast Fourier transform, Kalman filtering is probably the most important algorithmic technique ever devised. Papers on the subject have been filing numerous journals for decades. However, Kalman filtering is one of those rare topics that is not only popular in academic journals but also has a history of being rich in practical applications. Kalman filtering has been used in applications that include providing estimates for navigating the Apollo spacecraft, predicting short-term stock market fluctuations, and estimating user location with relatively inexpensive hand-held global positioning system receivers [32].

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.

3.1 The discrete Kalman filter

The Kalman filter addresses the general problem of trying to estimate the state $x \in R^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad \textbf{(3.1)}$$

With a measurement $z \in R^m$ that is

$$z_k = Hx_k + v_k \quad \textbf{(3.2)}$$

The random variable w_k and v_k represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions

$$p(w) \sim N(0, Q) \quad \textbf{(3.3)}$$

$$p(v) \sim N(0, R) \quad \textbf{(3.4)}$$

In practice, the process noise covariance Q and measurement noise covariance R matrices might change with each time step or measurement, however here we assume they are constant.

The $n \times n$ matrix A in the difference Equation 3.1 relates the state at the previous time step $k-1$ to the state at the current step k, in the absence of either a driving function or process noise. Note that in practice A might change with each time step, but here we assume it is constant. The $n \times l$ matrix B relates the optional control input $u \in R^l$ to the state x. The $m \times n$ matrix H in the measurement Equation 3.2

relates the state to the measurement z_k. In practice H might change with each time step or measurement, but here we assume it is constant.

3.1.1 The computational origin of the filter

We define $\hat{x}_k^- \in R^n$ (note the "super minus") to be our *a priori* state estimate at step k given knowledge of the process prior to step k, and $\hat{x}_k \in R^n$ to be our *a posteriori* state estimate at step k given measurement z_k. We can then define *a priori* and *a posteriori* estimate errors as

$$e_k^- \equiv x_k - \hat{x}_k^- \tag{3.5}$$

$$e_k \equiv x_k - \hat{x}_k \tag{3.6}$$

The *a priori* estimate error covariance is then

$$P_k^- = E[e_k^- e_k^{-T}] \tag{3.7}$$

And the *a posteriori* estimate error covariance is

$$P_k = E[e_k e_k^T] \tag{3.8}$$

In deriving the equations for the Kalman filter, we begin with the goal of finding an equation that computes an *a posteriori* state estimate $\hat{x}_k$ as a linear combination of an *a priori* estimate $\hat{x}_k^-$ and a weighted difference between an actual measurement z_k and a measurement prediction $H\hat{x}_k^-$ as shown below.

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \tag{3.9}$$

The difference $(z_k - H\hat{x}_k^-)$ is called the measurement innovation, or the residual. The residual reflects the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. A residual of zero means that the two are in complete agreement.

The $n \times m$ matrix K is chosen to be the gain or blending factor that minimizes the a posteriori error covariance. This minimization can be accomplished by first substituting Equation 3.9 into the above definition for e_k, substituting that into

Equation 3.8, performing the indicated expectations, taking the derivative of the trace of the result with respect to K, setting that result equal to zero, and then solving for K [33,34]. One form of the resulting K that minimizes Equation 3.8 is given by

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} = \frac{P_k^- H^T}{H P_k^- H^T + R} \tag{3.10}$$

Looking at Equation 3.10 we see that as the measurement error covariance R approaches zero, the gain K weights the residual more heavily. Specifically,

$$\lim_{R_k \to 0} K_k = H^{-1} \tag{3.11}$$

On the other hand, as the *a priori* estimate error covariance P_k^- approaches zero, the gain K weights the residual less heavily. Specifically,

$$\lim_{P_k^- \to 0} K_k = 0 \tag{3.12}$$

Another way of thinking about the weighting by K is that as the measurement error covariance R approaches zero, the actual measurement z_k is "trusted" more and more, while the predicted measurement $H\hat{x}_k^-$ is trusted less and less. On the other hand, as the *a priori* estimate error covariance P_k^- approaches zero the actual measurement z_k is trusted less and less, while the predicted measurement $H\hat{x}_k^-$ is trusted more and more.

3.1.2 The probabilistic origin of the filter

The justification for Equation 3.9 is rooted in the probability of the *a priori* estimate $\hat{x}_k^-$ conditioned on all prior measurements z_k (Bayes' rule). For now let it suffice to point out that the Kalman filter maintains the first two moments of the state distribution,

$$E[x_k] = \hat{x}_k \tag{3.13}$$

$$E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] = P_k \tag{3.14}$$

The *a posteriori* state estimate Equation 3.9 reflects the mean (the first moment) of the state distribution, it is normally distributed if the conditions of Equations 3.3 and

3.4 are met. The *a posteriori* estimate error covariance Equation 3.8 reflects the variance of the state distribution (the second non-central moment). In other words,

$$p(x_k|z_k) \sim N(E[x_k], E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]) = N(\hat{x}_k, P_k) \quad \textbf{(3.15)}$$

More details on the probabilistic origins of the Kalman filter can be found in [33, 34].

3.1.3 The discrete Kalman filter algorithm

We will begin this section with a broad overview, covering the "high-level" operation of one form of the discrete Kalman filter. After presenting this high-level view, we will narrow the focus to the specific equations and their use in this version of the filter.

The Kalman filter estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of (noisy) measurements. As such, the equations for the Kalman filter fall into two groups: *time update* equations and *measurement update* equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the *a priori* estimates for the next time step. The measurement update equations are responsible for the feedback, i.e. for incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate.

The time update equations can also be thought of as *predictor* equations, while the measurement updates equations can be thought of as *corrector* equations. Indeed the final estimation algorithm resembles that of a *predictor-corrector* algorithm for solving numerical problems.

The specific equations for the time and measurement updates are presented below.

- Discrete Kalman Filter Time Update Equations

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \quad \textbf{(3.16)}$$

$$P_k^- = AP_{k-1}A^T + Q \quad \textbf{(3.17)}$$

Again notice how the time update equations above project the state and covariance estimates forward from time step $k-1$ to step k. A and B are from Equation 3.1, while Q is from Equation 3.3.

- Discrete Kalman Filter Measurement Update Equations

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \quad \textbf{(3.18)}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad \textbf{(3.19)}$$

$$P_k = (I - K_k H) P_k^- \quad \textbf{(3.20)}$$

The first task during the measurement update is to compute the Kalman gain, K_k. Notice that the equation given here as Equation 3.18 is the same as Equation 3.10. The next step is to actually measure the process to obtain z_k, and then to generate an a posteriori state estimate by incorporating the measurement as in Equation 3.19. Again Equation 3.19 is simply Equation 3.9 repeated here for completeness. The final step is to obtain an a posteriori error covariance estimate via Equation 3.20.

After each time and measurement update pair, the process is repeated with the previous a posteriori estimates used to project or predict the new a priori estimates. This recursive nature is one of the very appealing features of the Kalman filter- it makes practical implementations much more feasible than (for example) an implementation of a Wiener filter [34] which is designed to operate on all of the data directly for each estimate. The Kalman filter instead recursively conditions the current estimate on all of the past measurements.

3.1.4 Filter parameters and tuning

In the actual implementation of the filter, the measurement noise covariance R is usually measured prior to operation of the filter. Measuring the measurement error covariance R is generally practical (possible) because we need to be able to measure the process anyway (while operating the filter) so we should generally be able to take some off-line sample measurement in order to determine the variance of the measurement noise.

The determination of the process noise covariance Q is generally more difficult as we typically do not have the ability to directly observe the process we are estimating.

Sometimes a relatively simple (poor) process model can produce acceptable results if one "injects" enough uncertainty into the process via the selection of Q. Certainly in this case one would hope that the process measurements are reliable.

In either case, whether or not we have a rational basis for choosing the parameters, often times superior filter performance (statistically speaking) can be obtained by *tuning* the filter parameters Q and R. The tuning is usually performed off-line, frequently with the help of another (distinct) Kalman filter in a process generally referred to as *system identification*.

In closing we note that under conditions where Q and R are in fact constant, both the estimation error covariance P_k and the Kalman gain K_k will stabilize quickly and then remain constant (see the filter update equations in Figure 3.2). if this is the case, these parameters can be pre-computed by either running the filter off-line, or for example by determining the steady-state value of P_k as described in [35].

It is frequently the case however that the measurement error (in particular) does not remain constant. Also, the process noise Q is sometimes changed dynamically during filter operation- becoming Q_k- in order to adjust to different dynamics. In such cases Q_k might be chosen to account for both uncertainties about the user's intentions and uncertainty in the model.

3.2 The extended Kalman filter (EKF)

3.2.1 The process to be estimated

As described above in section 3.1, the Kalman filter addresses the general problem of trying to estimate the state $x \in R^n$ of a discrete-time controlled process that is governed by a *linear* stochastic difference equation. But what happens if the process to be estimated and (or) the measurement relationship to the process is non-linear? Some of the most interesting and successful applications of Kalman filtering have been such situations. A Kalman filter that linearizes about the current mean and covariance is referred to as an *extended Kalman filter* or EKF.

In something akin to a Taylor series, we can linearize the estimation around the current estimate using the partial derivatives of the process and measurement functions to computes estimates even in the face of non-linear relationships. To do so, we must begin by modifying some of the material presented in section 3.1. Let us

assume that our process again has a state vector $x \in R^n$, but that the process is now governed by the *non-linear* stochastic difference equation

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \quad \textbf{(3.21)}$$

With a measurement $z \in R^m$ that is

$$z_k = h(x_k, v_k) \quad \textbf{(3.22)}$$

where the random variables w_k and v_k again represented the process and measurement noise as in Equations 3.3 and 3.4. In this case, the non-linear function f in the difference Equation 3.21 relates the state at the previous time step $k-1$ to the state at the current time step k. It concludes as parameters any driving function u_{k-1} and the zero-mean process noise w_k. The non-linear function h in the measurement

Equation 3.22 relates the state x_k to the measurement z_k.

In the practice of course one does not know the individual values of the noise w_k and v_k at each time step. However, one can approximate the state and measurement vector without them as

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0) \quad \textbf{(3.23)}$$

$$\hat{z}_k^- = h(\hat{x}_k^-, 0) \quad \textbf{(3.24)}$$

where $\hat{x}_k$ is some *a posteriori* estimate of the state (from a previous time step k).

It is important to note that a fundamental flaw of the EKF is that the distributions (or densities in the continuous case) of the various random variables are no longer normal after undergoing their respective nonlinear transformations. The EKF is simply an *ad hoc* state estimator that only approximates the optimality of Bayes' rule by linearization.

3.2.2 The computational origins of the filter

To estimate a process with non-linear difference and measurement relationships, we begin by writing new governing equations that linearize an estimate about Equations 3.23 and 3.24,

$$x_k \approx \hat{x}_k^- + A(x_{k-1} - \hat{x}_{k-1}) + Ww_{k-1} \quad \textbf{(3.25)}$$

$$z_k \approx \hat{z}_k^- + H(x_k - \hat{x}_k^-) + Vv_k \quad \textbf{(3.26)}$$

where

- x_k and z_k are the actual state and measurement vectors
- $\hat{x}_k^-$ and $\hat{z}_k^-$ are the approximate state and measurement vectors from Equations 3.23 and 3.24
- $\hat{x}_k$ is an *a posteriori* estimate of the state at step k
- The random variables w_k and v_k represent the process and measurement noise as in Equations 3.3 and 3.4
- A is the Jacobian matrix of partial derivatives of f with respect to x, that is

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_j}(\hat{x}_{k-1}, u_{k-1}, 0) \quad \textbf{(3.27)}$$

- W is the Jacobian matrix of partial derivatives of f with respect to w

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_j}(\hat{x}_{k-1}, u_{k-1}, 0) \quad \textbf{(3.28)}$$

- H is the Jacobian matrix of partial derivatives of h with respect to x

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_j}(\hat{x}_k^-, 0) \quad \textbf{(3.29)}$$

- V is the Jacobian matrix of partial derivatives of h with respect to v

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_j}(\hat{x}_k^-, 0) \quad \textbf{(3.30)}$$

Note that for simplicity in the notation we do not use the time step subscript k with the Jacobians

$A, W, H,$ and V, even though they are in fact difficult at each time step.

Now we define a new notation for the prediction error,

$$\hat{e}_{x_k}^- \equiv x_k - \hat{x}_k^- \tag{3.31}$$

and the measurement residual,

$$\hat{e}_{z_k}^- \equiv z_k - \hat{z}_k^- \tag{3.32}$$

Remember that in practice one does not have access to x_k in Equation 3.31, it is the *actual* state vector, i.e. the quantity one is trying to estimate. On the other hand, one *does* have access to z_k in Equation 3.32, it is the actual measurement that one is using to estimate x_k. Using Equations 3.31 and 3.32 we can write governing equations for an *error process* as

$$\hat{e}_{x_k}^- \approx A(x_{k-1} - \hat{x}_{k-1}) + \varepsilon_k \tag{3.33}$$

$$\hat{e}_{z_k}^- \approx h\hat{e}_{x_k}^- + \eta_k \tag{3.34}$$

where ε_k and η_k represent new independent random variables having zero mean and covariance matrices WQW^T and VRV^T, with Q and R as in Equations 3.3 and 3.4, respectively.

Notice that the Equations 3.33 and 3.34 are linear, and that they closely resemble the difference and measurement Equations 3.1 and 3.2 from the discrete Kalman filter. This motivates us to use the actual measurement residual $\hat{e}_{z_k}^-$ in Equation 3.32 and a second (hypothetical) Kalman filter to estimate the prediction error $\hat{e}_{x_k}^-$ given by Equation 3.33. This estimate, call it $\hat{e}_k$, could then be used along with Equation 3.31 to obtain the *a posteriori* state estimates for the original non-linear process as

$$\hat{x}_k = \hat{x}_k^- + \hat{e}_k \tag{3.35}$$

The random variables of Equations 3.33 and 3.34 have approximately the following probability distributions

$$p(\hat{e}_{x_k}^-) \sim N(0, E[\hat{e}_{x_k}^- \hat{e}_{x_k}^{-\,T}]) \tag{3.36}$$

$$p(\varepsilon_k) \sim N(0, WQ_kW^T) \tag{3.37}$$

$$p(\eta_k) \sim N(0, VRV^T) \tag{3.38}$$

Given these approximations and letting the predicted value of $\hat{e}_k$ be zero, the Kalman filter equation used to estimate $\hat{e}_k$ is

$$\hat{e}_k = K_k \hat{e}_{z_k}^- \tag{3.39}$$

By substituting Equation 3.39 back into Equation 3.35 and making use of Equation 3.32 we see that we do not actually need the second (hypothetical) Kalman filter:

$$\hat{x}_k = \hat{x}_k^- + K_k \hat{e}_{z_k}^- = \hat{x}_k^- + K_k(z_k - \hat{z}_k^-) \tag{3.40}$$

Equation 3.40 can now be used for the measurement update in the extended Kalman filter, with $\hat{x}_k^-$ and $\hat{z}_k^-$ coming from Equations 3.23 and 3.24, and the Kalman gain K_k coming from Equation 3.18 with the appropriate substitution for the measurement error covariance.

We now attach the subscript k to the Jacobians A, W, H, and V, to reinforce the notion that they are different at (and therefore must be recomputed at) each time step.

- EKF Time Update Equations

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{3.41}$$

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \tag{3.42}$$

As with the basic discrete Kalman filter, the time update equations above project the state and covariance estimates from the previous time step $k-1$ to the current time step k. Again f in Equation 3.41 comes from Equation 3.23, A_k and W_k are the process Jacobians at step *k*, and Q_k is the process noise covariance Equation 3.3 at step *k*.

- EKF Measurement Update Equations

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1} \tag{3.43}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{3.44}$$

$$P_k = (I - K_k H_k) P_k^- \quad \textbf{(3.45)}$$

As with the basic discrete Kalman filter, the measurement update equations above correct the state and covariance estimates with the measurement z_k. Again h in Equation 3.44 comes from Equation 3.24, H_k and V are the measurement Jacobians at step *k*, and R_k is the measurement noise covariance Equation 3.4 at step *k*. (Note we now subscript *R* allowing it to change with each measurement.)

An important feature of the EKF is that the Jacobian H_k in the equation for the Kalman gain K_k serves to correctly propagate or "magnify" only the relevant component of the measurement information. For example, if there is not a one-to-one mapping between the measurement z_k and the state via h, the Jacobian H_k affects the Kalman gain so that it only magnifies the portion of the residual $z_k - h(\hat{x}_k^-, 0)$ that does affect the state. Of course if over *all* measurements there is *not* a one-to-one mapping between the measurement z_k and the state via h, then as you might expect the filter will quickly diverge. In this case the process is *unobservable.*

3.3 Optimization via Kalman filter

Kalman filter is known to be an optimal estimator which its accurate estimation made it popular in comparison with other estimation methods. This feature of the filter can be utilized for the purpose of optimization. For this application of the Kalman filter, we need to define an evaluating function so that Kalman filter estimates the states while minimizing this evaluating function.

In a feedback control system where fuzzy PID-controller is used, we consider the controller and plant as a single nonlinear system that controller parameters are taken as the states and output of the plant as the measurement. Then an error-based evaluating function is defined which in this study it was considered as squared error between the output and reference for the membership function parameter optimization, and absolute error for the rule weight optimization. By defining this function, we can see the problem as optimization problem and apply the EKF to minimize error and consequently achieve the best membership functions or rule weights.

The nonlinear system model is as

$$x_{m+1} = x_m + w_m \tag{3.46}$$

$$d_m = E_m + v_m \tag{3.47}$$

E_m is the error of the system having the membership function parameters or rule weights at mth iteration, and vectors w_m and v_m are artificially added noises to increase the stability of the parameter estimation algorithm.

Comparing equations 3.46 and 3.47 with equations 3.21 and 3.22, A, W and V matrices are identity matrices but we need to calculate H Jacobian matrix. After this linearization, EKF algorithm can be applied to optimize the parameters of the controller to have the least error.

4. OPTIMIZATION OF MEMBERSHIP FUNCTION PARAMETERS USING EXTENDED KALMAN FILTER

A great deal of work has been done on optimization of a fuzzy logic controller, especially fuzzy PID-type controllers. As mentioned before, different parameters of these controllers have been the topic of research in the literature. However, most researches have been focused on shaping the membership functions since fuzzy systems show high sensitivity on the parameters of membership functions. Many successful applications of both derivative-based and derivative-free optimization methods to tune the membership function parameters have been reported in the literature [8, 11, 36].

One of the most critical disadvantages of derivative-based optimization methods is that they are applicable to only specific performance indices or types of membership functions because of complicated and time consuming calculations. Extended Kalman Filter optimization method is of this type which using it, so far, only (input and output) triangular membership functions of a fuzzy logic controller have been optimized [37]. In this study, we extend this approach to optimize a special type of membership functions, called rational-powered membership functions, which are extended form of triangular/trapezoidal types. Furthermore, output singleton membership functions are widely used in fuzzy logic systems because software and specifically hardware realization of these functions is easier that of triangular/trapezoidal membership functions; therefore we provide required formulas for output singleton optimization via the extended Kalman filter.

4.1 Standard fuzzy PID controller

In this thesis, we will deal with standard fuzzy PID controllers, formed using a fuzzy PD controller with an integrator and a summation unit at the output, as it is shown in Figure 4.1.

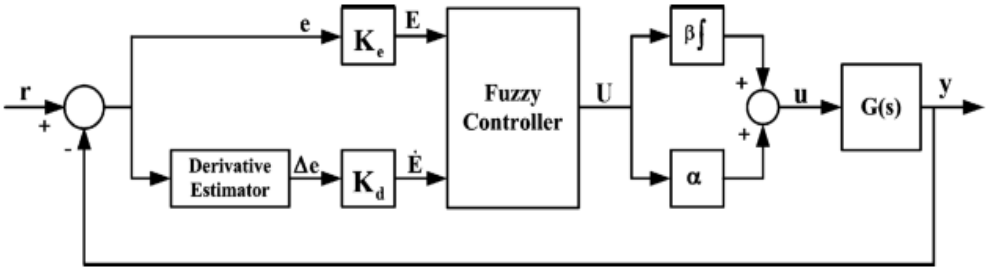


Figure 4.1 : Closed-loop control structure for PID-type FLC.

The output of the fuzzy PID controller is given by

$$u = \alpha U + \beta \int U dt \qquad \textbf{(4.1)}$$

The error (e) and the derivative of error ($\dot{e}$) are used as the inputs and the change of the control signal (U) is used as the output of the FLC. The input scaling factors k_e for error and k_d for the derivative of error normalize the inputs to the range in which the membership functions of the inputs are defined. Universe of discourse for both of the inputs is taken to be $[-1, 1]$. For each input variables, five rational-powered membership functions are requested to use, as illustrated in subsection 4.2.

The following classical symmetrical rule table shown in Table 4.1 is used in this study.

Table 4.1 : Rule base for fuzzy PID-type controller.

x_1	x_2				
	NB	NS	ZE	PS	PB
NB	NB	NB	NS	NS	ZE
NS	NB	NS	NS	ZE	PS
ZE	NS	NS	ZE	PS	PS
PS	NS	ZE	PS	PS	PB
PB	ZE	ZE	PS	PB	PB

In practice, fuzzy control is applied using local inferences. That means each rule is inferred and the results of the inferences of individual rules are then aggregated. The most common inference methods are: the max-min method, the max-product method and the sum-product method, where the aggregation operator is denoted by either max or sum, and the fuzzy implication operator is denoted by either min or product. Especially the max-min calculus of fuzzy relations offers a computationally nice and

expressive setting for constraint propagation. Finally, a defuzzification method is needed to obtain a crisp output from the aggregated fuzzy result. Popular defuzzification methods include maximum matching and centriod defuzzification. The centroid defuzzification is widely used for fuzzy control problems where a crisp output is needed, and maximum matching is often used for pattern matching problems where we need to need to know the output class. In this thesis, the fuzzy reasoning results of outputs are gained by aggregation operation of fuzzy sets of inputs and designed fuzzy rules, where max-min aggregation method and cetroid defuzzification method are used.

4.2 Rational-powered membership functions

Rational-powered membership functions are extended form of triangular/trapezoidal membership functions. Figure 4.2 shows typical shapes of these kind of membership functions.

Note that the formulas for a trapezoidal membership functions can be obtained easily from a triangular membership functions, and the derivative formulas of the fuzzy output with respect to membership function parameters are zero when the input is in the core of the membership function (the area which the membership values are 1); therefore, to avoid further complexity, we eliminate the discussion of trapezoidal membership functions.

Triangular membership function has two straight lines, but rational-powered membership function is composed of two curves. The ith rational-powered membership function is uniquely determined by its five parameters: modal point (c_i), upper half-width (b_i^+) and its power (a_i^+), and lower half-width (b_i^-) and its power (a_i^-). The degree of membership of the jth crisp input ($z_j, j = 1, 2$) in its ith fuzzy set is obtained by the following equation:

$$f_{ji}(z_j) = \begin{cases} (1 + \frac{z_j - c_{ij}}{b_{ij}^-})^{a_{ij}^-} & if\ c_{ij} - b_{ij}^- \le z_j \le c_{ij} \\ (1 - \frac{z_j - c_{ij}}{b_{ij}^+})^{a_{ij}^+} & if\ c_{ij} \le z_j \le c_{ij} + b_{ij}^+ \\ 0 & otherwise \end{cases} \quad \textbf{(4.2)}$$

where $a_{ij} > 0$. Special case is when $a_{ij}^{+} = a_{ij}^{-} = a_{ij} = 1$; in this case the curves are reduced to straight lines and the membership functions are triangular. Therefore three parameters remain for triangular functions which are modal point, and upper and lower half-widths. Furthermore, if we see the fuzzy system as a black box, and only consider its input to output characteristic, we can say that modifying conventional membership functions by applying linguistic hedges results in the same effect of defining a rational-powered membership function with appropriate power; hence rational-powered membership functions can be interpreted as the extended form of those functions which are generated by applying linguistic hedges. For example, applying hedge "very" to a triangular membership function results in a function which has the same effect of a rational-powered membership function with equal upper and lower half-width powers of $a_{ij} = 2$.

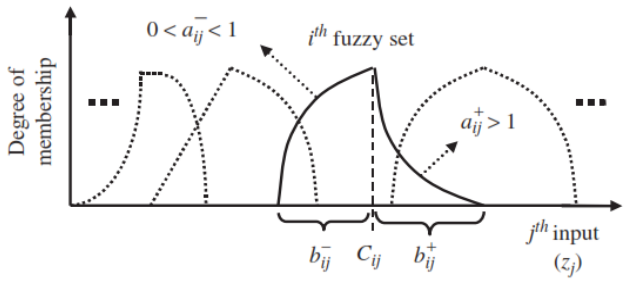


Figure 4.2 : Rational-powered membership functions.

4.3 Application of extended Kalman filter to membership function tuning

We apply the Extended Kalman filter algorithm to optimize the membership function parameters of a fuzzy system. We will try to tune the membership function parameters in an on-line manner.

4.3.1 Mathematics of fuzzy operations

Assume that the consequent of the kth rule, $m_k(\hat{y})$, is a singleton placed at Γ_u; therefore this rule has the following form:

IF z_1 *is fuzzy set* A_1, ***and*** z_2 *is fuzzy set* B_2, ***then*** *output* $(m(\hat{y}))$ *is* Γ_u.

Consequently the contribution of this rule in the fuzzy output is

$$\bar{m}_k(\hat{y}) = \omega_k m_k(\hat{y}) = \omega_k \Gamma_u \qquad \textbf{(4.3)}$$

where ω_k is the activation level of the kth rule:

$$\omega_k = min\left[f_{A_1}(z_1), f_{B_2}(z_2)\right] \tag{4.4}$$

If the system has a total of M rules, the aggregated fuzzy output is

$$m(\hat{y}) = \sum_{k=1}^{M} \bar{m}_k(\hat{y}) \tag{4.5}$$

This fuzzy output is mapped to a crisp output using the weighted average defuzzification:

$$\hat{y} = \frac{m(\hat{y})}{\sum_{k=1}^{M} \omega_k} = \frac{\sum_{k=1}^{M} \omega_k m_k(\hat{y})}{\sum_{k=1}^{M} \omega_k} \tag{4.6}$$

4.3.2 On-line tuning of membership functions via EKF

Derivations of the extended Kalman Filter were discussed in section 2 in details. For completeness we briefly outline the algorithm and show how it can be applied to fuzzy membership function parameters optimization. For clear notation, the scalars, column vectors, and matrices are referred by lower case letters, bold-face lower case letters and upper case letters, respectively. Consider a nonlinear discrete time system which is described as

$$\boldsymbol{x}_{m+1} = \boldsymbol{f}(\boldsymbol{x}_m) + \boldsymbol{w}_m \tag{4.7}$$

$$\boldsymbol{d}_m = \boldsymbol{h}(\boldsymbol{x}_m) + \boldsymbol{v}_m \tag{4.8}$$

where vector $\boldsymbol{x}_m$ is the state of the system at iteration m (i.e. mth state estimation), and $\boldsymbol{d}_m$ is the observation (or measurement) vector; $\boldsymbol{f}(.)$ and $\boldsymbol{h}(.)$ are nonlinear vector functions and $\boldsymbol{w}$ and $\boldsymbol{v}$ are process and observation noises, respectively. The initial state of the system ($\boldsymbol{x}_0$) and sequences $\{\boldsymbol{w}_m\}$ and $\{\boldsymbol{v}_m\}$ are Gaussian and independent from each other with

$$E(\boldsymbol{x}_0) = \bar{\boldsymbol{x}}_0 \tag{4.9}$$

$$E[(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0)(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0)^T] = \boldsymbol{P}_0 \tag{4.10}$$

$$E(\boldsymbol{w}_m) = 0 \tag{4.11}$$

$$E(\boldsymbol{w}_m \boldsymbol{w}_l^T) = \boldsymbol{Q}\delta_{ml} \tag{4.12}$$

$$E(\boldsymbol{v}_m) = 0 \tag{4.13}$$

$$E(\boldsymbol{v}_m \boldsymbol{v}_l^T) = \boldsymbol{R}\delta_{ml} \tag{4.14}$$

where $E(.)$ is the expectation operator; $\boldsymbol{P}$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the covariance matrices of the state estimation error, process and observation noises, respectively; and finally δ_{ml} is the Kronecker delta:

$$\delta_{ml} = \begin{cases} 1 & if\ m = l \\ 0 & otherwise \end{cases} \tag{4.15}$$

The goal is to find an estimate $\hat{\boldsymbol{x}}_m$ of $\boldsymbol{x}_m$ given $\boldsymbol{d}_0, \ldots, \boldsymbol{d}_{m-1}$. It can be shown that the desired estimate $\hat{\boldsymbol{x}}_m$ is obtained by the recursive extended Kalman Filter:

$$F_m = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{f}(\hat{\boldsymbol{x}}_{m-1})} \tag{4.16}$$

$$H_m = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{f}(\hat{\boldsymbol{x}}_{m-1})} \tag{4.17}$$

$$K_m = P_m H_m^T (R + H_m P_m H_m^T)^{-1} \tag{4.18}$$

$$\hat{\boldsymbol{x}}_m = f(\hat{\boldsymbol{x}}_{m-1}) + K_m[\,\boldsymbol{d}_{m-1} - \boldsymbol{h}(\hat{\boldsymbol{x}}_{m-1})] \tag{4.19}$$

$$P_{m+1} = F_m(P_m - K_m H_m P_m)F_m^T + Q \tag{4.20}$$

where $P_1 = F_0 P_0 F_0^T + Q$, and K_m is known as Kalman gain.

In this stage, we consider the fuzzy PID-type controller along with the plant as a single nonlinear system which the parameters of the membership functions are taken as the states of this system. And the output of the whole system is aquired as the measurement. Extended Kalman filter is applied to this nonlinear system and tries to estimate the parameters of the controller (states of the system) while minimizing the evaluating function.

Considering the output of the fuzzy system ($\hat{y}_n$) and knowing its desired value (y_n), we can define an error-based function as

$$E = \frac{1}{2} {E_n}^2 \tag{4.21}$$

$$E_n = y_{ref} - y \tag{4.22}$$

Now we need to construct the state vector for the Kalman filtering. We let the parameters of the membership functions constitute the state of a nonlinear system, and we let the output of the fuzzy system constitute the output of the nonlinear system to which the extended Kalman filter is applied.

As shown in the previous subsection, we consider a fuzzy system which has five fuzzy sets for the first input, five fuzzy sets for the second input, and five singletons for the output. Since each input fuzzy set has five parameters and output membership function has one parameter, the state vector has 55 elements overall. The state of the nonlinear system can then be represented as

$$\boldsymbol{x} = [c_{11}\ b_{11}^{+}\ b_{11}^{-}\ a_{11}^{+}\ a_{11}^{-} \dots\ c_{51}\ b_{51}^{+}\ b_{51}^{-}\ a_{51}^{+}\ a_{51}^{-} \dots\ c_{12}\ b_{12}^{+}\ b_{12}^{-}\ a_{12}^{+}\ a_{12}^{-} \dots\ c_{52}\ b_{52}^{+}\ b_{52}^{-}\ a_{52}^{+}\ a_{52}^{-} \dots\ \Gamma_1\ \Gamma_2 \dots\ \Gamma_5] \tag{4.23}$$

The nonlinear model is given as

$$\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + \boldsymbol{w}_m \tag{4.24}$$

$$\boldsymbol{d}_m = E_m + \boldsymbol{v}_m \tag{4.25}$$

where w_n and v_n are artificially added noise processes. Comparing Equations 4.24 and 4.25 with equations 4.7 and 4.8, $\boldsymbol{f}(.)$ is the identity mapping, E_m is the error of the system having the rule weights at mth iteration. Now we can project the time update and measurement update of the extended Kalman filter in order to estimate the next state of the system.

$$F_m = I \tag{4.26}$$

$$H_m = \frac{\partial E}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{m-1}} \tag{4.27}$$

$$K_m = P_m H_m^T (R + H_m P_m H_m^T)^{-1} \tag{4.28}$$

$$\hat{\boldsymbol{x}}_m = \hat{\boldsymbol{x}}_{m-1} + K_m(\boldsymbol{d}_{m-1} - E_{m-1}) = \hat{\boldsymbol{x}}_{m-1} - K_m E_{m-1} \tag{4.29}$$

$$P_{m+1} = P_m - K_m H_m P_m + Q \tag{4.30}$$

where I is the identity matrix and $P_1 = P_0 + Q$. Matrix H_m is calculated in the following subsection. It is assumed that Q and R do not change at each iteration. If the state of the filter has μ elements, and the system has k outputs, then $P_0 = p_0 I_{\mu\times\mu}$, $Q = q_0 I_{\mu\times\mu}$ and $R = r_0 I_{k\times k}$. Initial covariance p_0, q_0, and r_0 are the tuning parameters of the algorithm and usually are chosen by the trial and error to have the best convergence. The optimization algorithm is illustrated in Figure 4.3.

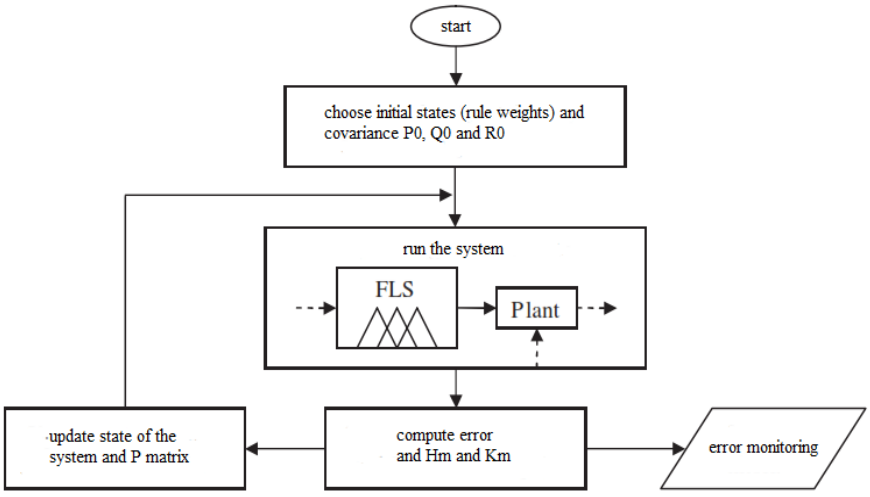


Figure 4.3 : Flowchart of the EKF-based optimization.

4.3.3 Derivative formulas

The partial derivative of error-based function (E) with respect to state parameters is calculated. Matrix H is a vector of fifty five elements:

$$H = \left[\frac{\partial E}{\partial c_{11}} \frac{\partial E}{\partial b_{11}^+} \frac{\partial E}{\partial b_{11}^-} \frac{\partial E}{\partial a_{11}^+} \frac{\partial E}{\partial a_{11}^-} \dots \frac{\partial E}{\partial \Gamma_5}\right] \tag{4.31}$$

The elements of H are obtained by straightforward calculus and algebra. Since input membership functions can affect the output of fuzzy system (and consequently error-

based function) through activation level (ω), only those parameters which determine ω in Equation 4.4 have nonzero derivatives. Therefore we define function $r_{ijk} = 1$, if ith fuzzy set of jth input determine the activation level of the kth rule (ω_k); and if not so, we define $r_{ijk} = 0$. This function is used to obtain the derivative formulas with respect to input MF parameters. By using the chain rule for derivation we have:

- Input Modal Points:

$$\frac{\partial E}{\partial c_{ij}} = \frac{\partial E}{\partial E_n}\frac{\partial E_n}{\partial y}\frac{\partial y}{\partial \hat{y}_n}\frac{\partial \hat{y}_n}{\partial \omega_k}\frac{\partial \omega_k}{\partial c_{ij}} \tag{4.32}$$

Now we calculate each term separately:

$$\frac{\partial E}{\partial E_n} = E_n \tag{4.33}$$

$$\frac{\partial E_n}{\partial y} = -1 \tag{4.34}$$

$$\frac{\partial y}{\partial \hat{y}_n} = \frac{y(t) - y(t-1)}{\hat{y}_n(t) - \hat{y}_n(t-1)} \tag{4.35}$$

$$\frac{\partial \hat{y}_n}{\partial \omega_k} = \frac{m_k(y)\sum_{l=1}^{M}\omega_l - \sum_{l=1}^{M}\omega_l m_l(y)}{(\sum_{l=1}^{M}\omega_l)^2} = \frac{m_k(y) - \hat{y}_n}{\sum_{l=1}^{M}\omega_l} \tag{4.36}$$

$$\frac{\partial \omega_k}{\partial c_{ij}} = \frac{\partial f_{ij}(z_j)}{\partial c_{ij}} r_{ijk} \tag{4.37}$$

$$\frac{\partial f_{ij}(z_j)}{\partial c_{ij}} = \begin{cases} \dfrac{a_{ij}^-}{c_{ij} - b_{ij}^- - z_j} f_{ij}(z_j) & if\ c_{ij} - b_{ij}^- < z_j < c_{ij} \\ \dfrac{a_{ij}^+}{c_{ij} + b_{ij}^+ - z_j} f_{ij}(z_j) & if\ c_{ij} < z_j < c_{ij} + b_{ij}^+ \\ 0 & otherwise \end{cases} \tag{4.38}$$

- Input Half-widths: for upper half-widths, we have

$$\frac{\partial E}{\partial {b_{ij}}^+} = \frac{\partial E}{\partial E_n}\frac{\partial E_n}{\partial y}\frac{\partial y}{\partial \hat{y}_n}\frac{\partial \hat{y}_n}{\partial {b_{ij}}^+} \tag{4.39}$$

The first three terms is calculated in the previous part, thus we have

$$\frac{\partial \hat{y}_n}{\partial {b_{ij}}^+} = \sum_{k=1}^{M} \frac{\partial \hat{y}_n}{\partial \omega_k} \frac{\partial \omega_k}{\partial {b_{ij}}^+} \tag{4.40}$$

$$\frac{\partial \omega_k}{\partial {b_{ij}}^+} = \frac{\partial f_{ij}(z_j)}{\partial {b_{ij}}^+} r_{ijk} \tag{4.41}$$

$$\frac{\partial f_{ij}(z_j)}{\partial {b_{ij}}^+} = \begin{cases} a_{ij}^+ \left(\frac{1}{b_{ij}^+ + c_{ij} - z_j} - \frac{1}{b_{ij}^+} \right) f_{ij}(z_j) & if\ c_{ij} < z_j < c_{ij} + b_{ij}^+ \\ 0 & otherwise \end{cases} \tag{4.42}$$

For lower half-widths, Equations 4.39-4.41 are valid by changing ${b_{ij}}^+$ to ${b_{ij}}^-$. Finally we have

$$\frac{\partial f_{ij}(z_j)}{\partial {b_{ij}}^-} = \begin{cases} a_{ij}^- \left(\frac{1}{b_{ij}^- - c_{ij} + z_j} - \frac{1}{b_{ij}^-} \right) f_{ij}(z_j) & if\ c_{ij} - b_{ij}^- < z_j < c_{ij} \\ 0 & otherwise \end{cases} \tag{4.43}$$

- Input Powers: for upper half-width powers, the following results are obtained:

$$\frac{\partial E}{\partial {a_{ij}}^+} = \frac{\partial E}{\partial E_n} \frac{\partial E_n}{\partial y} \frac{\partial y}{\partial \hat{y}_n} \frac{\partial \hat{y}_n}{\partial {a_{ij}}^+} \tag{4.44}$$

As before, we have

$$\frac{\partial \hat{y}_n}{\partial {a_{ij}}^+} = \sum_{k=1}^{M} \frac{\partial \hat{y}_n}{\partial \omega_k} \frac{\partial \omega_k}{\partial {a_{ij}}^+} \tag{4.45}$$

$$\frac{\partial \omega_k}{\partial {a_{ij}}^+} = \frac{\partial f_{ij}(z_j)}{\partial {a_{ij}}^+} r_{ijk} \tag{4.46}$$

$$\frac{\partial f_{ij}(z_j)}{\partial {a_{ij}}^+} = \begin{cases} \frac{1}{a_{ij}^+} f_{ij}(z_j) \ln\left(f_{ij}(z_j) \right) & if\ c_{ij} < z_j < c_{ij} + b_{ij}^+ \\ 0 & otherwise \end{cases} \tag{4.47}$$

For lower half-width powers, Equations 4.44-4.46 are valid by changing ${a_{ij}}^+$ to ${a_{ij}}^-$, and

$$\frac{\partial f_{ij}(z_j)}{\partial a_{ij}^{-}} = \begin{cases} \frac{1}{a_{ij}^{-}} f_{ij}(z_j) \ln\left(f_{ij}(z_j)\right) & if\ c_{ij} - b_{ij}^{-} < z_j < c_{ij} \\ 0 & otherwise \end{cases} \quad \textbf{(4.48)}$$

- Output membership function parameter

$$\frac{\partial E}{\partial \Gamma_l} = \frac{\partial E}{\partial E_n} \frac{\partial E_n}{\partial y} \frac{\partial y}{\partial \hat{y}_n} \frac{\partial \hat{y}_n}{\partial \Gamma_l} \quad \textbf{(4.49)}$$

$$\frac{\partial \hat{y}_n}{\partial \Gamma_l} = \frac{\sum_{k=1}^{M} \omega_k s_{lk}}{\sum_{k=1}^{M} \omega_k} \quad \textbf{(4.50)}$$

4.4 Simulation results

Conventional fuzzy PID-type controllers show almost acceptable performance while applied to linear systems and need to tune the controller in these plants is not critical. Therefore we project the proposed method to time-delay systems, and increase the time delay of the system step by step in order to check the performance of the tuned controller. To be fair, the same number of rational-powered input membership functions and output singletons are used for each of these fuzzy PID controllers. The rule table is the same as the table given in previous subsection.

4.4.1 System I

We start with a small delay of 0.1. The system is given as

$$G(s) = \frac{2e^{-0.1s}}{s^2 + 3s + 2} \quad \textbf{(4.51)}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.25,\ \alpha = 0.2$ and $\beta = 1.6$. The unit step response of these Fuzzy PID controllers are shown in Figure 4.4.

4.4.2 System II

We increase the time delay to 0.3. The system is given as

$$G(s) = \frac{2e^{-0.3s}}{s^2 + 3s + 2} \quad \textbf{(4.52)}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.25,\ \alpha = 0.2$ and $\beta = 1.6$. The unit step response of these Fuzzy PID controllers are shown in Figure 4.5.

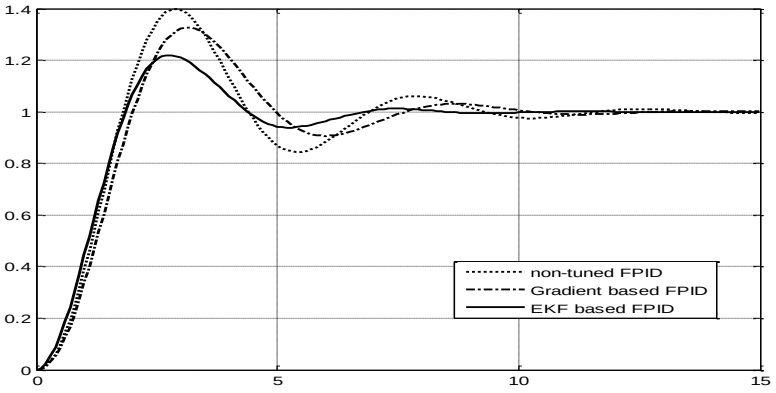


Figure 4.4 : Closed-loop unity responses for system I.

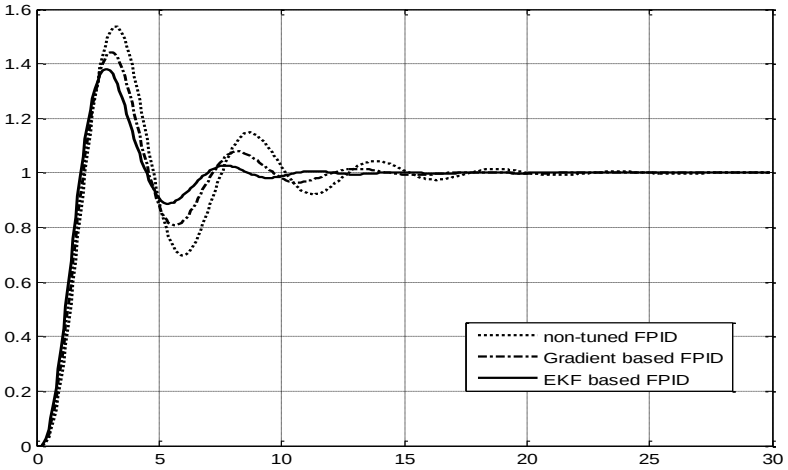


Figure 4.5 : Closed-loop unity responses for system II.

4.4.3 System III

If we increase the time delay to 0.8, we have

$$G(s) = \frac{2e^{-0.8s}}{s^2 + 3s + 2} \qquad \textbf{(4.53)}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.25,\ \alpha = 0.2$ and $\beta = 0.7$. Figure 4.6 shows the performance of the proposed tuning method in comparison with the other fuzzy PID-type controllers.

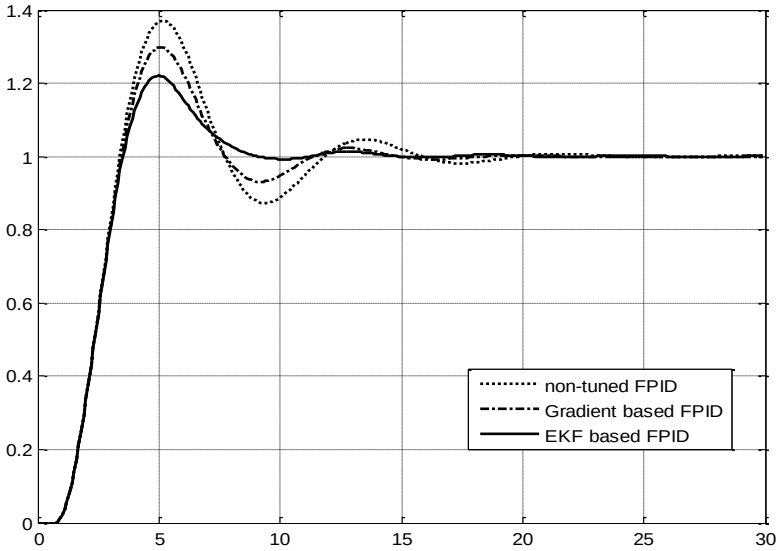


Figure 4.6 : Closed-loop unity responses for system III.

4.4.4 System IV

In this case the time delay of the system is 1.1 second. The system is

$$G(s) = \frac{2e^{-1.1s}}{s^2 + 3s + 2} \qquad \textbf{(4.54)}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.1,\ \alpha = 0.2$ and $\beta = 0.5$. Figure 4.7 shows the performance of the proposed tuning method in comparison with the other fuzzy PID-type controllers.

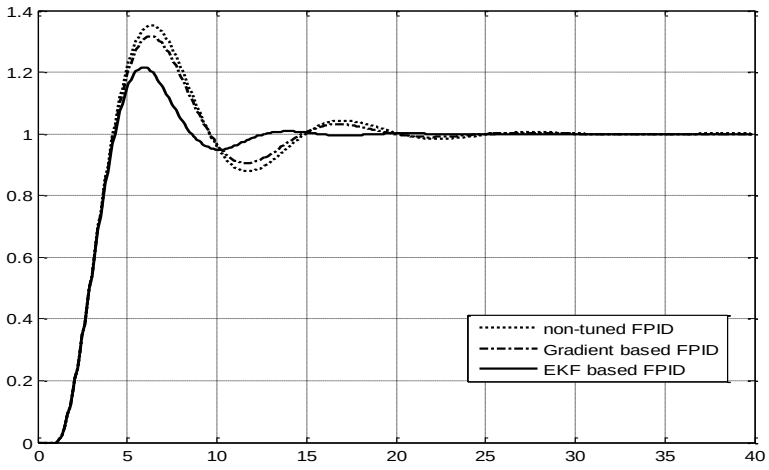


Figure 4.7 : Closed-loop unity responses for system IV.

4.4.5 System V

In this case the time delay of the system is 1.5 second. The system is

$$G(s) = \frac{2e^{-1.5s}}{s^2 + 3s + 2} \tag{4.55}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.1,\ \alpha = 0.2$ and $\beta = 0.4$. Figure 4.8 shows the performance of the proposed tuning method in comparison with the other fuzzy PID-type controller.

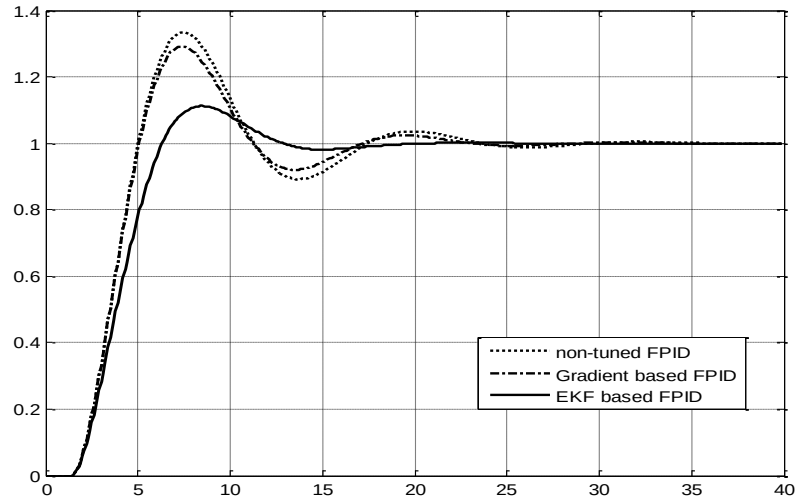


Figure 4.8 : Closed-loop unity responses for system V.

In order to evaluate the performance of the proposed tuning method in comparison with the non-tuned and gradient descent based fuzzy PID-type controllers, we make use of thefollowing performance measures: namely maximum overshoot (%OS), settling time (T_s), integral time absolute error (ITAE) and integral absolute error (IAE).

a. Integral Absolute Error (IAE) which is defined as:

$$IAE = \int_0^\infty |r(t) - y(t)|dt \tag{4.56}$$

b. Integral Time Absolute Error (ITAE) which is defined as:

$$ITAE = \int_0^\infty t|r(t) - y(t)|dt \tag{4.57}$$

As it is obvious from Table 4.2, maximum overshoot in all the five systems has decreased remarkably. In addition, the system tracks the reference in a small settling time while being controlled by EKF-based fuzzy PID-type controller. However, rise time stays the same or even in the fifth system it increased slightly. The other two error-based performance measures show the effectiveness of the proposed method, for they indicate a smaller value.

4.5 Noise effect on the system

In this subsection we add a continuous noise of 10% of the reference to the control signal, i.e. in the range of $[-0.1,0.1]$. The following simulations show the efficiency of the proposed method in handling the noise. As it is obvious from the Figures 4.9 and 4.10, the EKF-based fuzzy PID-type controllers can track the reference with a small deviation.

Table 4.2 : Performance measures for three controllers for different time delays.

system	%OS			$T_s(s)$			ITAE			IAE		
	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID
I	40	32.7	21.9	8.3	6.9	5.7	55.2	46.8	25	22.9	21.8	16.6
II	53.8	44.4	38	12	9	6.4	138.4	74.5	44.4	34.7	26.6	21.6
III	37	29.8	22.1	11	10.1	7.5	146	98.4	74.1	39.46	34.5	31.2
IV	35.2	31.6	21.5	13.9	13.4	8	219	181.6	101	48	45	37.3
V	33.3	29.2	11.2	16.2	15.3	11	274.8	221.6	139.6	55	51.2	45.4

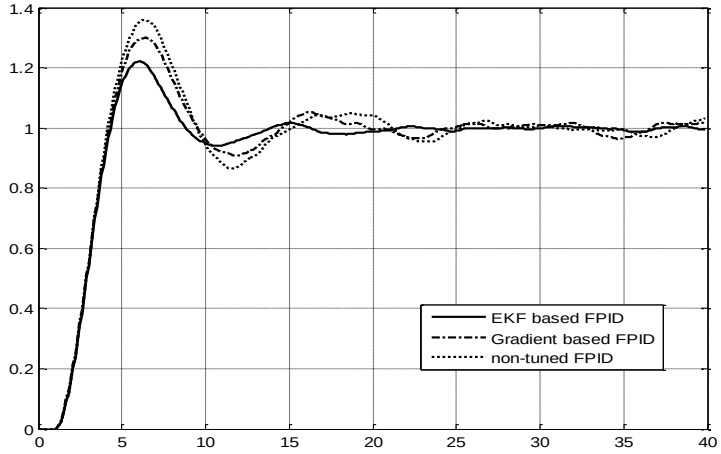


Figure 4.9 : Unity responses for system IV in presence of noise.

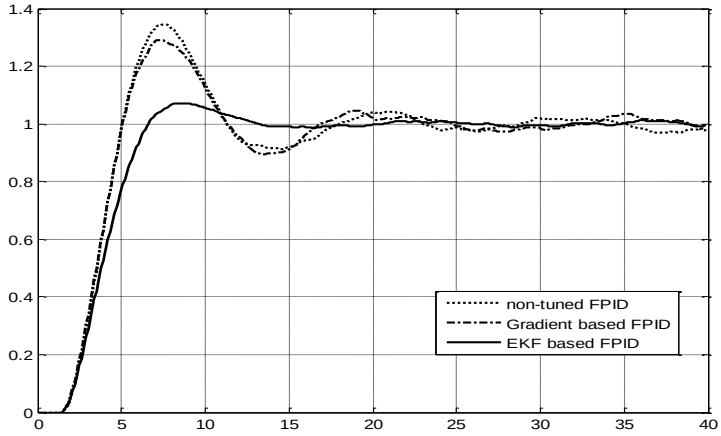


Figure 4.10 : Unity responses for system V in presence of noise.

5. OPTIMIZATION OF RULE WEIGHTS USING EXTENDED KALMAN FILTER

Fuzzy rule weights optimization is an area that has attracted interest of researchers in recent years. Fuzzy rule weights as structural parameters changes the importance and contribution degree of the rule with respect to current inputs. In literature there have been numerous tuning aproaches for rule weights.

Teng, Xiang, Wang, and Wu [9] proposed a genetic weighted fuzzy rule based system in which the parameters of membership functions including position and shape of the fuzzy rule set and weights of the rules are evolved using a genetic algorithm. Alternatively, Genc, Yesil, Eksin, Guzelkaya, and Tekin [10] proposed a fuzzy rule base shifting scheme for systems with time delay to improve system performance. The shifting scheme of the fuzzy rules is tabulated with respect to the normalized dead time, therefore, in some way, the structural parameters of the FLC was tuned in an on-line manner.

Karasakal et al. [38] performed the adjustment of rule weight values in a simple way using the absolute value function of the normalized system error directly as the weight values. Another method was presented in [39] where the rule weight optimization was done in a way using a fuzzy logic mechanism which uses the normalized acceleration in addition to system error as inputs and the output of this fuzzy mechanism is directly assigned as the fuzzy rule weight value of the fuzzy rules.

In this study, a new method is proposed for the adjustment of the fuzzy rule weights of the fuzzy PID controllers in an on-line manner. For the analysis, the state space nonlinear error model of the fuzzy system along with the plant is defined while the rule weights of the fuzzy rules are taken as the states of this model. Then the extended Kalman filter is applied to this nonlinear model to update the states in each time step. The effectiveness of the proposed self tuning algorithm is demonstrated on linear and non-linear systems by simulations.

5.1 The fuzzy rule weights

A comprehensive analysis on the fuzzy rule weights can be found in [40]. The weight (w) is a real value variable that can change for each rule and it is generally added to a rule with the phrase ‘‘with w’’. The weight values can be considered as a structural parameter of a fuzzy PID controller. The weight of the rule shows the importance or influence of that fuzzy rule for the inputs at a specified time. The fuzzy rules of FLCs with weights are generally written as;

IF the error (e) is N_k and the derivative of error ($\dot{e}$) is M_k THEN the change of control signal (U) is C_k with W_k.

Generally the weights are applied to the rules in two different ways:

a. The weights are applied to the complete rule. In this case, the degree of fulfillment of the rule is changed by the weight.

b. The weights are applied to the consequent of the rules. In this case, the output of the rule is changed with the weight.

When the singleton membership functions are used at the output part of the rules, the rule weights occur in the sums of the numerator and denominator for case (a). The output of FLC is calculated as;

$$u = \frac{\sum_i^m f_k c_k \omega_k}{\sum_i^m f_k \omega_k} \qquad \textbf{(5.1)}$$

For case (b), the rule weights occur in the nominator of the above equation, so the output of FLC is calculated as;

$$u = \frac{\sum_i^m f_k c_k \omega_k}{\sum_i^m f_k} \qquad \textbf{(5.2)}$$

where f_k, ω_k and c_k are the firing strength, weight and the output singleton membership value of k^{th} rule, respectively.

In this study, due to simplicity of obtaining the derivation of output control signal with respect to rule weights case ‘a’ is chosen to be implemented.

5.2 Theory of rule weight optimization via extended Kalman filter

Similar to the previous section, first we extract the error model of the whole system (i.e. controller plus plant) that EKF is to be applied to. The rule weights of the fuzzy controller are taken as the states and output of the system is taken as the measurement. After that EKF is applied to this nonlinear system to tune the rule weights while minimizing the evaluating function.

For notational convenience, we consider a two-input one-output fuzzy system (which can be extended to arbitrary number of inputs and outputs). In general, we can define an evaluating function to describe the performance of the system. For example, considering the output of the fuzzy system ($\hat{y}_n$) and knowing its desired value (y_n), we can define an error-based function as

$$E = abs(E_n) = |E_n| \tag{5.3}$$

$$E_n = y_n - \hat{y}_n \tag{5.4}$$

As the behavior of the system closes to the desired manner, the error (E) becomes smaller, and therefore the performance is better.

By defining this error-based evaluating function for a fuzzy system, we can see the optimization of the system as an optimization problem and we can apply the extended Kalman filter to minimize error (and consequently achieve the optimum rule weights). For this purpose, we determine the state of the system and the observation vector. In order to cast the rule weight optimization problem in a form suitable for Kalman filtering, we let the rule weights of the rule base constitute the state of a nonlinear system, and we let the output of the fuzzy system constitute the output of the nonlinear system to which the extended Kalman filter is applied.

The fuzzy PID-type controller that is used in this section in exactly similar to the controller in the section 4 except for the triangular symmetrical input membership function shown in figure 5.1. Since the fuzzy system has five fuzzy sets for the first input and five fuzzy sets for the second input, there are 25 rule weights for the rules of the rule base.

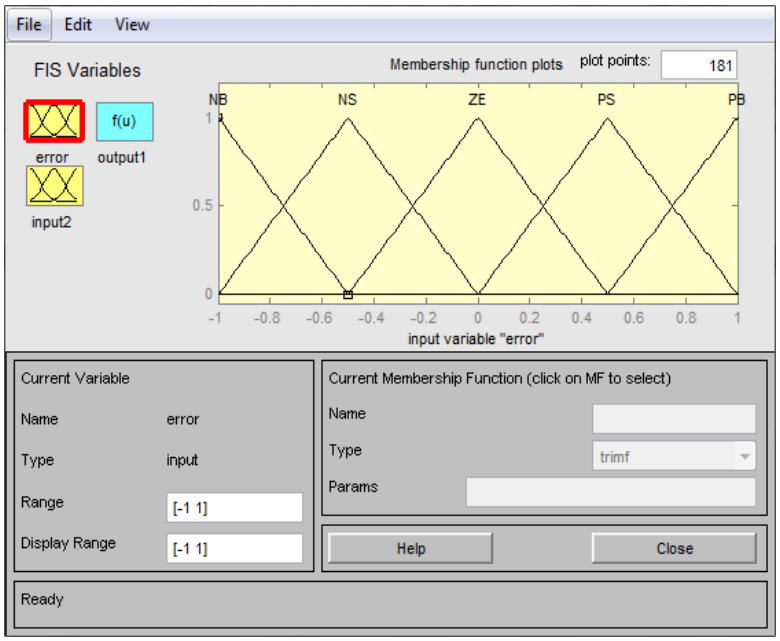


Figure 5.1 : Symmetric triangular membership functions.

The state of the nonlinear system can then be represented as

$$\boldsymbol{x} = [\omega_1\ \omega_2\ \omega_3\ \ldots\ \omega_{24}\ \omega_{25}]^T \tag{5.5}$$

The vector x thus consists of the entire fuzzy rule weights arranged in a linear array. The nonlinear system model to which the Kalman filter can be applied is

$$\boldsymbol{x}_{m+1} = \boldsymbol{x}_m \tag{5.6}$$

$$\boldsymbol{d}_m = \boldsymbol{h}(x_m) \tag{5.7}$$

where $h(x_n)$ is the fuzzy system's nonlinear mapping between the rule weights and the single output of the fuzzy system. In order to execute a stable Kalman filter algorithm, we need to add some artificial process noise and measurement noise to the system model. This is similar to the approach taken for neural network training using Kalman filters [41]. So we rewrite Equations 5.6 and 5.7 as

$$\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + \boldsymbol{w}_m \tag{5.8}$$

$$\boldsymbol{d}_m = E_m + \boldsymbol{v}_m \tag{5.9}$$

where w_n and v_n are artificially added noise processes. Comparing Equations 5.8 and 5.9 with equations extended Kalman filter equations, $\boldsymbol{f}(.)$ is the identity mapping, E_m is the error of the system having the rule weights at mth iteration.

Since we want the actual output of the fuzzy system to be the same as the desired output, the desired value of the observation vector (E) is $\boldsymbol{d} = 0$. Consequently the extended Kalman filter recursion can be applied as

$$F_m = I \tag{5.10}$$

$$H_m = \frac{\partial E}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{m-1}} \tag{5.11}$$

$$K_m = P_m H_m^T (R + H_m P_m H_m^T)^{-1} \tag{5.12}$$

$$\hat{\boldsymbol{x}}_m = \hat{\boldsymbol{x}}_{m-1} + K_m(\boldsymbol{d}_{m-1} - E_{m-1}) = \hat{\boldsymbol{x}}_{m-1} - K_m E_{m-1} \tag{5.13}$$

$$P_{m+1} = P_m - K_m H_m P_m + Q \tag{5.14}$$

where I is the identity matrix and $P_1 = P_0 + Q$. Matrix H_m is calculated in the following subsection.

5.2.1 Derivative formulas

In this subsection, the partial derivative of error-based function (E) with respect to state parameters is calculated. Because the system we used has five fuzzy sets for each input, thus twenty five rules overall, matrix H is a vector of twenty five elements:

$$H = \left[\frac{\partial E}{\partial \omega_1}\ \frac{\partial E}{\partial \omega_2}\ \frac{\partial E}{\partial \omega_3}\ \frac{\partial E}{\partial \omega_4} \ldots \frac{\partial E}{\partial \omega_{24}}\ \frac{\partial E}{\partial \omega_{25}}\right] \tag{5.15}$$

These formulas are used in the extended Kalman filter recursion to estimate the state of the system. The elements of H are obtained by straightforward calculus and algebra. By using the chain rule for derivation we have:

$$\frac{\partial E}{\partial \omega_k} = \frac{\partial E}{\partial E_n}\frac{\partial E_n}{\partial y_n}\frac{\partial y_n}{\partial u_c}\frac{\partial u_c}{\partial \omega_k} \tag{5.16}$$

where

$$\frac{\partial E}{\partial E_n} = \frac{E_n}{|E_n|} \tag{5.17}$$

$$\frac{\partial E_n}{\partial y_n} = 1 \tag{5.18}$$

$$\frac{\partial y_n}{\partial u_c} = \frac{y_n(t) - y_n(t-1)}{u_c(t) - u_c(t-1)} \tag{5.19}$$

Now we obtain the derivative formula of control signal u_c with respect to the rule weights for both cases *a* and *b*:

$$\frac{\partial u_c}{\partial \omega_k} = \frac{f_k c_k (\sum_i^m f_k \omega_k) - f_k (\sum_i^m f_k c_k \omega_k)}{(\sum_i^m f_k \omega_k)^2} \tag{5.20}$$

$$\frac{\partial u_c}{\partial \omega_k} = \frac{f_k c_k}{\sum_i^m f_k} \tag{5.21}$$

In the next subsection, we apply the above mentioned method to tune the fuzzy PID controller for both a linear and time-delay system.

5.3 Simulation results

During the simulations, the conventional fuzzy PID controller with no rule weight adjustment will be denoted as non-tuned FPID controller, fuzzy PID that is tuned by Gradient Descent method as Gradiend-based FPID and fuzzy PID controller that is tuned by the proposed method is consider as EKF-based tuned FPID controller. To be fair, the same number and shape for input and output membership functions given in previous subsections are used for each of these fuzzy PID controllers.

5.3.1 System I

Firstly we try the method on a second order linear plant given as below:

$$G(s) = \frac{2}{s^2 + 3s + 2} \tag{5.22}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.1,\ \alpha = 0.1$ and $\beta = 1.8$.

These scaling factors lead to an oscillatory output response with an overshoot. The unit step output responses of these Fuzzy PID controllers are shown in Figure 5.2.

5.3.2 System II

Since most of the high-order processes can usually be modeled as second order plus dead time (SOPDT) linear systems, the following system is taken into consideration:

$$G(s) = \frac{2e^{-1.5s}}{s^2 + 3s + 2} \qquad \textbf{(5.23)}$$

The scaling factor values are taken as $k_e = 1, k_d = 0.1,\ \alpha = 0.4$ and $\beta = 1.8$. The unit step output responses of these Fuzzy PID controllers are shown in Figure 5.3.

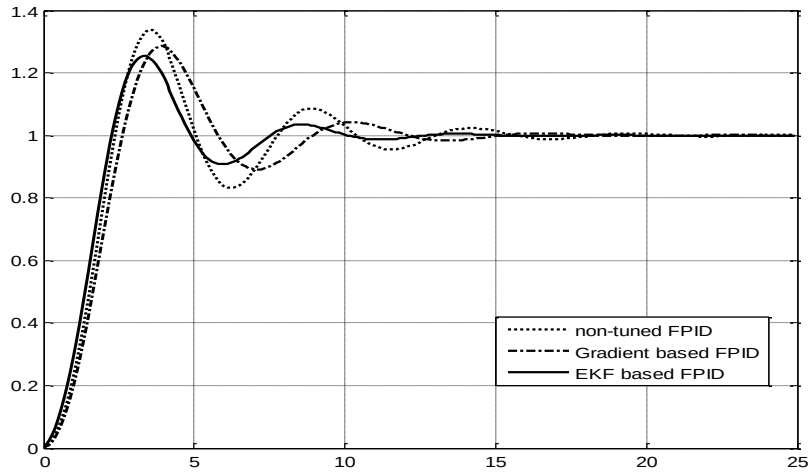


Figure 5.2 : Closed-loop unity responses for system I.

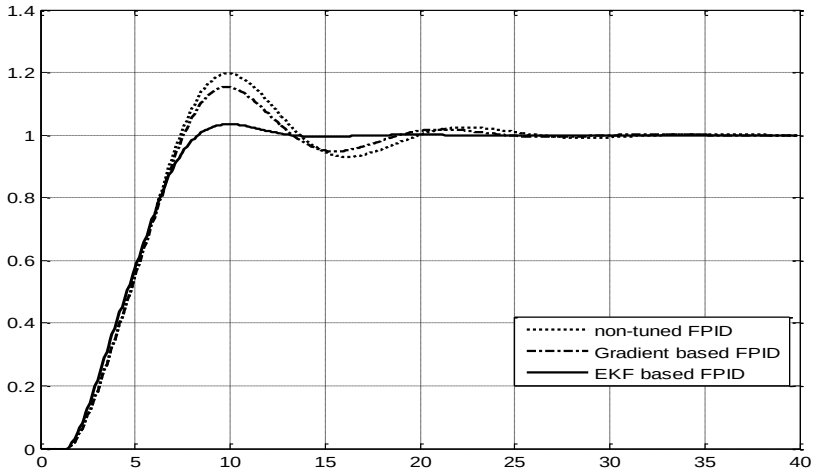


Figure 5.3 : Closed-loop unity responses for system II.

5.4 Noise effect on the systems

Since real systems are always exposed to noise, we artificially add noise of 10% to the control signal. Noise value varies in the range of $[-0.1, 0.1]$, because reference is set to 1 in all simulations. Being known to be an optimal estimator in noisy environments, Kalman filter, as the tuning part of fuzzy PID-type controller, copes with the noise well. The following figures show the effectiveness of the proposed method for both systems. All the parameters of the controller are kept the same as previous section.

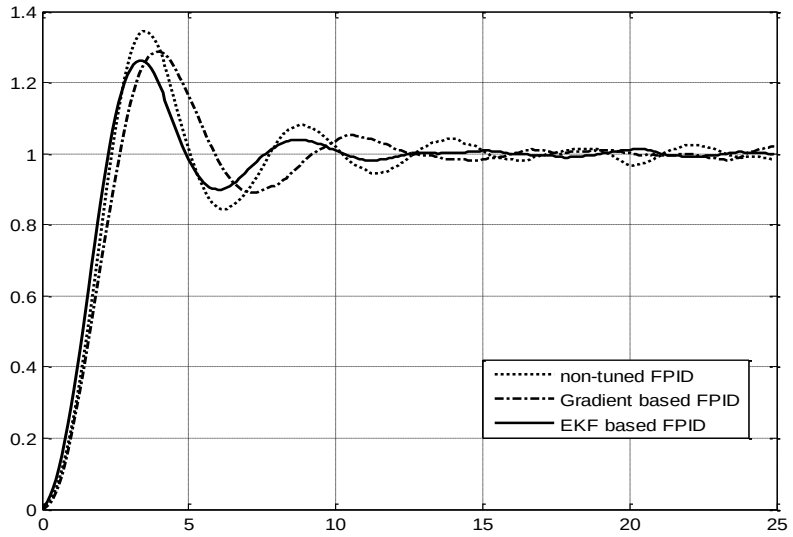


Figure 5.4 : Unity responses for system I in presence of noise.

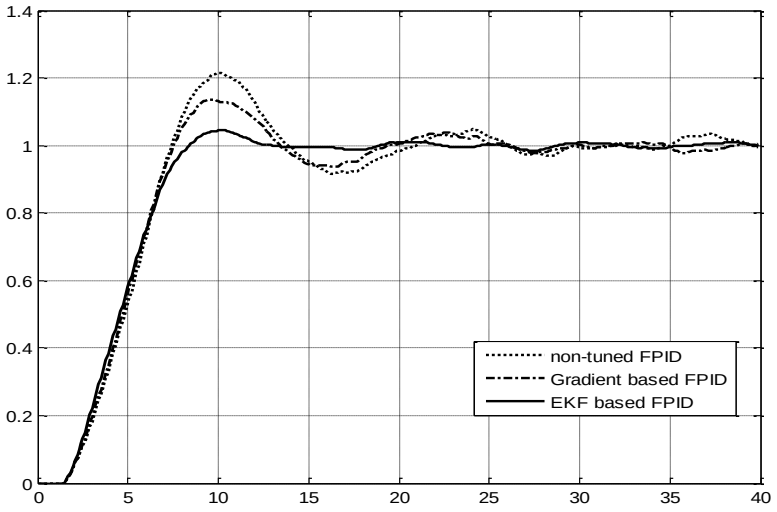


Figure 5.5 : Unity responses for system II in presence of noise.

In order to compare the performance of the proposed extended Kalman filter based rule weight tuning method with gradient descent based controller and non-tuned controller, four different performance measures are considered: namely maximum overshoot (%OS), settling time (T_s), integral time absolute error (ITAE) and integral absolute error (IAE). The performance analysis of the fuzzy logic controllers according to the performance indices is given in Table 5.1. The proposed extended Kalman filter based fuzzy PID controller reduces the overshoot value as it is expected when it is compared to the other fuzzy PID controllers. As given in Figures 5.2 and 5.3, all of the controllers have almost the same rise time, but the settling time of the proposed controller is significantly smaller than the others. The ITAE and IAE value extended Kalman filter based fuzzy controller is drastically less than the other fuzzy PID controllers since the oscillations vanish.

Table 5.1 : Performance measures for three controllers.

system s	%OS			$T_s(s)$			ITAE			IAE		
	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID	Non-tuned FPID	Grad-based FPID	EKF-based FPID
$\frac{2}{s^2+3s+2}$	33.6	28.6	25.5	9.7	8.3	6.8	87.91	57.75	45.41	27.69	20.25	21.44
$\frac{2e^{-1.5s}}{s^2+3s+2}$	19.6	15.4	3.5	17.8	16.1	7.6	445.23	207.95	140.32	91.3	53.34	48.1

6. CONCLUSIONS AND RECOMMENDATIONS

In this study, an on-line tuning method for optimization of both structural and tuning parameters, namely rule weights and membership function parameters, of fuzzy logic controllers based on extended Kalman filter is proposed. Extended Kalman filter provides a powerful mathematical tool to deal with the nonlinearity of the system and also reduction of the effect of disturbance. The proposed tuning algorithm was applied to constant time-delay second order system in presence of noise and compared with the response of standard non-tuned and gradient descent based fuzzy PID-type controllers.

Firstly, we applied the extended Kalman filter based optimization method to tune the membership function parameters, namely right and left half-widths, right and left hedges, and modal points. These parameters can have a great influence on the performance of the fuzzy logic controller. In this case, tuning algorithm estimates the next states of the controller parameters so that it minimizes the cost function which is squared error between the system response and reference. It cannot decrease the rise time value, but overshoot and settling time reduces remarkably. Also the value of ITAE and IAE is noticeably low in comparison with other FPID responses. Also simulations illustrate that the proposed method could successfully deal with the continuous noise projected on the system and track the reference with a small error.

Other study was projecting the extended Kalman filter to tune the rule weights of the fuzzy PID-type controller. The extended Kalman filter changes the rule weights of the rule base in order to minimize the performance index, which is the absolute error between the system response and reference. The performance of the proposed tuning method is evaluated according to four performance measures tabulated in Table 5.1. Percent of overshoot and settling time decrease considerably while rise time value stays almost the same. In addition, two error-based performance indices, namely ITAE and IAE, show a good improvement in the performance of fuzzy PID-type controller. Moreover, the proposed on-lined tuned fuzzy PID-type controller could handle the noise of 10% successfully, while other FPID controllers were inefficient.

For future work, the proposed method may be applied to an experimental setup to examine the efficiency of it when projected to a real system. Also another area can be extending this approach to variant time-delay systems, since most real systems are of this type.

REFERENCES

[1] **Mamdani, E. H.** (1974). Application of fuzzy algorithm for simple dynamic plant. Proc. IEEE, 1585-1588.

[2] **Mamdani, E. H.** (1993). Twenty years of fuzzy control: experiences gained and lessons learnt. Proc. 2nd IEEE inc. conf. fuzzy systems, San Francisco, CA, 339-344.

[3] **Gurbuz, E.** (2007). Self tuning PID parameters using fuzzy logics vs nonlinear controllers. Master's Thesis, Istanbul Technical University.

[4] **Jantzen, J.** (2007). Foundation of Fuzzy Control. Wiley.

[5] **Chung, H. Y., Chen, B. C., and Lin, J. J.** (1998). A PI-type fuzzy controller with self-tuning scaling factors. Fuzzy Sets and Systems, 93, 23–28.

[6] **Mudi, R. K., and Pal, N. R.** (1999). A robust self-tuning scheme for PI- and PD-type fuzzy controllers. IEEE Transactions on Fuzzy Systems, 7(1), 2–16.

[7] **Juang, Y. T., Chang, Y. T., & Huang, C. P.** (2008). Design of fuzzy PID controllers using modified triangular membership functions. Information Sciences, 178, 1325–1333.

[8] **Ahn, K. K., & Truong, D. Q.** (2009). Online tuning fuzzy PID controller using robust extended Kalman filter. Journal of Process Control, 19, 1011–1023.

[9] **Teng, M., Xiang, F., Wang, R., & Wu, Z.** (2004). Using genetic algorithm for weighted fuzzy rule-based system. In Proceedings of the 5th world congress on intelligent control and automation, pp. 4292–4295.

[10] **Genc, H. M., Yesil, E., Eksin, I., Guzelkaya, M., & Tekin, O. A.** (2009). A rule base modification scheme in fuzzy controllers for time-delay systems. Expert Systems with Applications, 36, 8476–8486.

[11] **Simon, D.** (2002). Training Fuzzy Systems with the Extended Kalman Filter. Fuzzy Sets and Systems, 132, 189-199.

[12] **Surmann, H.** (1996). Genetic optimization of a fuzzy system for charging batteries. IEEE Trans. Industrial Electron. 43, 541–548.

[13] **Figueiredo, M., Gomide, F.** (1999). Design of fuzzy systems using neurofuzzy networks. IEEE Trans. Neural Networks 10, 815–827.

[14] **Magdalena, L., Monasterio-Huelin, F.** (1997). Fuzzy logic controller with learning through the evolution of its knowledge base. Internat. J. Approx. Reasoning 16, 335–358.

[15] **Smith, S., Comer, D.** (1991). Automated calibration of a fuzzy logic controller using a cell state space algorithm. IEEE Control Systems Magazine 11, 18–28.

[16] **Wu, R., Chen, S.** (1999). A new method for constructing membership functions and fuzzy rules from training examples. IEEE Trans. Systems Man Cybernet.—Part B 29, 25–40.

[17] **Tao, C., Taur, J.** (1999). Design of fuzzy controllers with adaptive rule insertion, IEEE Trans. Systems Man Cybernet.—Part B: Cybernet. 29, 389–397.

[18] **Simon, D.** (2000). Design and rule base reduction of a fuzzy filter for the estimation of motor currents. Internat. J. Approx. Reasoning 25, 145–167.

[19] **Simon, D.** (2002). Sum normal optimization of fuzzy membership functions. Internat. J. Uncertainty Fuzziness Knowledge-Based Systems 10 (4), 363–384.

[20] **Sugeno, M., Tanaka, K.** (1991). Successive identification of a fuzzy model and its applications to prediction of a complex system. Fuzzy Sets and Systems 42, 315–334.

[21] **Wang, L., Mendel, J.** (1992). Back-propagation of fuzzy systems as nonlinear dynamic system identifiers. IEEE Internat. Conf. on Fuzzy Systems, San Diego, CA, pp. 1409–1418.

[22] **Chen, Y. Y., Yen, C. C.** (1992). PD-type vs PID-type Fuzzy Controllers. IEEE Region 10 Conference, Tencon 92, Melbourne, Australia, 11-13 November.

[23] **Wang, L., Yen, J.** (1998). Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and Kalman filter. Fuzzy Sets and Systems 101, 353–362.

[24] **Ramaswamy, P., Riese, Mr., Edwards, R., Lee, K.** (1993). Two approaches for automating the tuning process of fuzzy logic controllers. IEEE Conf. on Decision and Control, San Antonio, TX, pp. 1753–1758.

[25] **Jang, J. S. R., Sun, C. T., and Mizutani, E.** (1997). Neuro-fuzzy and Soft Computing Matlab Curriuculum Series. Prentice Hall.

[26] **Lee, C. C.** (1990). Fuzzy logic in control systems: Fuzzy logic controller. IEEE Transactions on System, Man, and Cybernetics, 20 (2), 404-435.

[27] **Qiao, W. Z., and Mizumoto, M.** (1996). PID Type Fuzzy Controller and Parameters Adaptive Method. Fuzzy Sets and Systems, 78, 23-35.

[28] **Güzelkaya, M., Eksin, I., and Yeşil, E.** (2003). Self-tuning of PID-type Fuzzy Logic Controller Coefficient via Relative Rate Observer. Engineering Application of Artificial Intelligence, 16, 227-236.

[29] **Galichet, S., Foulloy, L.** (1995). Fuzzy controllers: synthesis and equivalences. IEEE Transactions on Fuzzy Systems, 3, 140-148.

[30] **Huang, T. T., Chung, H. Y., and Lin, J. J.** (1999). A Fuzzy PID Controller Being Like Parameter Varying PID. IEEE International Fuzzy Systems Conference Proceeding, Vol. 1. Seoul, Korea, pp. 269–275.

[31] **Kalman, R. E.** (1960). A new approach to linear filtering and prediction problems. Journal of Basic Engineering, Vol. 82, No. 1, pp. 35-46.

[32] **Zarchan, P., Musoff, H.** (2005). Foundation of Kalman Filtering: A Practical Approach. Second Edition. Progress in Astronautics and Aeronautics, volume 208.

[33] **Maybeck, P. S.** (1979). Stochastic Models, Estimation, and Control. Volume 1, Academic Press, Inc.

[34] **Brown, R. G., and Hwang, P. Y. C.** (1992). Introduction to Random Signals and Applied Kalman Filtering. Second Edition, John Wiley & Sons, Inc.

[35] **Grewal, M. S., and Angus, P. A.** (1993). Kalman Filtering Theory and Practice. Upper Saddle River, NJ USA, Prentice Hall.

[36] **Kashtiban, M. M., Khoei, A., and Hadidi, Kh.** (2008). Optimization of Rational-powered Membership Functions Using Extended Kalman Filter. Fuzzy Sets and Systems, 159, 3232-3244.

[37] **Hu, B., Mann, G. K. I., Gosine, R. G.** (1999). A New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers. IEEE Transactions on Fuzzy Systems 7 (5), 521-539.

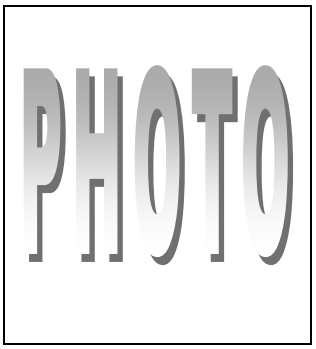
[38] **Karasakal, O., Guzelkaya, M., Eksin, I., Yesil, E.** (2011). An error-based on-line rule weight adjustment method for fuzzy PID controllers. Expert Systems with Applications, 38, 10124-10132.

[39] **Karasakal, O., Guzelkaya, M., Eksin, I., Yesil, E., Kumbasar, T.** (2013). Online tuning of fuzzy PID controllers via rule weighing based on normalized acceleration. Engineering Applications of Artificial Intelligence. 26, 184-197.

[40] **Nauck, D., Kruse, R.** (1998). How the learning of rule weights affects the interpretability of fuzzy systems. In: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZIEEE'98), pp. 1235–1240.

[41] **Puskorius, G., Feldkamp, L.** (1994). Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks. IEEE Trans. Neural Networks 5, 279–297.

CURRICULUM VITAE



Name Surname: Nasser Arghavani

Place and Date of Birth: Tebriz/1984

E-Mail: nasser.arghavani@gmail.com arghavanin@itu.edu.tr

B.Sc.: Electronics Engineering

M.Sc. (If exists): Control and Automation Engineering